

Modelling Concurrency with Comtraces and Generalized Comtraces

Ryszard Janicki^{*,1}

Department of Computing and Software, McMaster University, Hamilton, ON, L8S 4K1 Canada

Dai Tri Man Lê²

Department of Computer Science, University of Toronto, Toronto, ON, M5S 3G4 Canada

Abstract

Comtraces (*combined traces*) are extensions of Mazurkiewicz traces that can model the “not later than” relationship. In this paper, we first introduce the novel concept of *generalized comtraces*, extensions of comtraces that can additionally model the “non-simultaneously” relationship. Then we study some basic algebraic properties and canonical forms of both comtraces and generalized comtraces. Finally we analyze the relationship between (generalized) comtraces and (generalized) stratified order structures in detail. The major technical contributions of this paper are the results showing that generalized comtraces and generalized stratified order structures can uniquely represent one another.

Key words: generalized trace theory, trace monoids, step sequences, stratified partial orders, stratified order structures, canonical representations

1. Introduction

Mazurkiewicz traces, or just traces³, are quotient monoids over sequences (or words) [2, 23, 5]. The theory of traces has been utilized to tackle problems from quite diverse areas including combinatorics, graph theory, algebra, logic and especially concurrency theory [5].

As a language representation of finite partial orders, traces can sufficiently model “true concurrency” in various aspects of concurrency theory. However, some aspects of concurrency cannot be adequately modelled by partial orders (cf. [9, 11]), and thus cannot be modelled in terms of traces. For example, neither traces nor partial orders can model the “not later than” relationship [11]. If an event a is performed “not later than” an event b , then this “not later than”

^{*}Corresponding author

Email addresses: janicki@mcmaster.ca (Ryszard Janicki), ledt@cs.toronto.edu (Dai Tri Man Lê)

¹Partially supported by NSERC of Canada Grant.

²Partially supported by Ontario Graduate Scholarship.

³The word “trace” has many different meanings in Computer Science and Software Engineering. In this paper, the word “trace” is reserved for *Mazurkiewicz trace* [5, 23] and this is different from “traces” used in Hoare’s CSP [8] and Parnas’ Trace Assertion Method [1].

relationship can be modelled by the following set of two step sequences $\mathbf{x} = \{\{a\}\{b\}, \{a,b\}\}$; where *step* $\{a,b\}$ denotes the simultaneous performance of a and b and the step sequence $\{a\}\{b\}$ denotes the execution of a followed by b . But the set \mathbf{x} cannot be represented by any trace (or equivalently any partial order), even if the generators, i.e. elements of the trace alphabet, are sets and the underlying monoid is the monoid of step sequences (as in [29]).

To overcome those limitations the concept of a *comtrace* (*combined trace*) was introduced in [12]. Comtraces are finite sets of equivalent step sequences and the congruence is determined by a relation *ser*, which is called *serializability* and in general is not symmetric. Monoid generators are ‘steps’, i.e., finite sets, so they have *some internal structure* that can be used to define equations that generate the quotient monoid. Set union is used to define comtrace congruence. Comtraces provide a formal language counterpart to *stratified order structures* (*so-structures*) and were used to provide a semantics of Petri nets with inhibitor arcs. The paper [12] contains a major result showing that every comtrace uniquely determines a so-structure, yet contains very little algebraic theory of comtraces, and the reciprocal relationship, how a finite so-structure determines an appropriate comtrace, is not discussed at all. We will discuss this reciprocal relationship in detail as well as a formal relationship between traces and comtraces.

A so-structure [6, 10, 12, 13] is a triple (X, \prec, \sqsubseteq) , where \prec and \sqsubseteq are binary relations on X . They were invented to model both the “earlier than” (the relation \prec) and the “not later than” (the relation \sqsubseteq) relationships, under the assumption that all system runs are modelled by stratified partial orders, i.e., step sequences. They have been successfully applied to model inhibitor and priority systems, asynchronous races, synthesis problems, etc. (see for example [12, 18, 17, 20] and others). However, so far comtraces are used much less often than so-structures, even though in many cases they appear to be more natural than so-structures. Perhaps this is due to the lack of a sufficiently developed theory of quotient monoids for comtraces analogous to that of traces.

However, neither comtraces nor so-structures can adequately model the “non-simultaneously” relationship, which could be defined by the set of step sequences $\{\{a\}\{b\}, \{b\}\{a\}\}$ with the additional assumption that the step $\{a,b\}$ is not allowed. In fact, both comtraces and so-structures can adequately model concurrent histories only when paradigm π_3 of [11, 13] is satisfied. Intuitively, paradigm π_3 formalizes the class of concurrent histories satisfying the condition that if both $\{a\}\{b\}$ and $\{b\}\{a\}$ belong to the concurrent history, then so does $\{a,b\}$ (i.e., all of these step sequences $\{a\}\{b\}$, $\{b\}\{a\}$ and $\{a,b\}$ are equivalent observations).

To model the general case that includes the “non-simultaneously” relationship, we need the concept of *generalized stratified order structures* (*gso-structures*), which were introduced and analyzed in [7, 9]. A gso-structure is a triple $(X, \diamond, \sqsubseteq)$, where \diamond and \sqsubseteq are binary relations on X modelling the “non-simultaneously” and the “not later than” relationships respectively under the assumption that all system runs are modelled by stratified partial orders. In this paper, we propose a language counterpart of gso-structures, called *generalized comtraces* (*g-comtraces*). We will analyze in detail the properties of g-comtraces, their canonical representations, and most importantly the formal relationship between g-comtraces and gso-structures.

This paper is the expansion and revision of our results from [16, 22]. The content of the paper is organized as follows. In the next section, we review some basic concepts of order theory and monoid theory. Section 3 recalls the concept of Mazurkiewicz traces and discusses its relationship to finite partial orders. Section 4 surveys some basic background on the relational structures model of concurrency [6, 10, 12, 13, 7, 9]. Comtraces are defined and their relationship to traces is discussed in Section 5, and the g-comtraces are introduced in Section 6. Various basic algebraic properties of both comtrace and g-comtrace congruences are discussed in Section 7. Section 8 is devoted to canonical representations of traces, comtraces and g-comtraces.

A few properties of both comtrace and g-comtrace languages are presented in Section 9. In Section 10 we discuss both the so-structures defined by comtraces and the comtraces generated by so-structures. The gso-structures generated by g-comtraces are defined and analyzed in Section 11, and the g-comtraces generated by gso-structures are defined and analyzed in Section 12. Concluding remarks are made in Section 13. We also include two Appendixes containing some long and technical proofs of results from Section 11.

2. Orders, Monoids, Sequences and Step Sequences

In this section, we recall some standard notations, definitions and results which are used extensively in this paper.

2.1. Relations, Orders and Equivalences

Let X be a set. The *powerset* of X will be denoted by $\mathcal{P}(X)$, i.e. $\mathcal{P}(X) \stackrel{df}{=} \{Y \mid Y \subseteq X\}$, and the set of all *non-empty* subsets of X , i.e., $\widehat{\mathcal{P}}(X) \stackrel{df}{=} \mathcal{P}(X) \setminus \{\emptyset\}$, will be denoted by $\widehat{\mathcal{P}}(X)$.

We let id_X denote the *identity relation* on a set X , i.e., $id_X \stackrel{df}{=} \{(x, x) \mid x \in X\}$. If R and S are binary relations on a set X (i.e., $R, S \subseteq X \times X$), then their *composition* $R \circ S$ is defined as $R \circ S \stackrel{df}{=} \{(x, y) \in X \times X \mid \exists z \in X. (x, z) \in R \wedge (z, y) \in S\}$. We also define

$$R^0 \stackrel{df}{=} id_X \quad R^i \stackrel{df}{=} R^{i-1} \circ R \quad (\text{for } i \geq 1) \quad R^+ \stackrel{df}{=} \bigcup_{i \geq 1} R^i \quad R^* \stackrel{df}{=} \bigcup_{i \geq 0} R^i$$

The relations R^+ and R^* are called the (*irreflexive*) *transitive closure* and *reflexive transitive closure* of R respectively.

A binary relation $R \subseteq X \times X$ is an *equivalence relation* on X if and only if (iff) the following must hold for all $a, b, c \in X$: $a R a$ (*reflexive*), $a R b \Rightarrow b R a$ (*symmetric*) and $a R b \wedge b R c \Rightarrow a R c$ (*transitive*). If R is an equivalence relation, then for every $x \in X$, the set $[x]_R \stackrel{df}{=} \{y \mid y R x \wedge y \in X\}$ is the equivalence class of x with respect to R . We also define $X/R \stackrel{df}{=} \{[x]_R \mid x \in X\}$, i.e., the set of all equivalence classes of X with respect to R . We drop the subscript and write $[x]$ to denote the equivalence class of x when R is clear from the context.

A binary relation $\prec \subseteq X \times X$ is a (*strict*) *partial order* iff for all $a, b, c \in X$, we have: $\neg(a \prec a)$ (*irreflexive*) and $a \prec b \wedge b \prec c \Rightarrow a \prec c$ (*transitive*). The pair (X, \prec) in this case is called a *partially ordered set* (also called a *poset*), i.e., the set X is partially ordered by the relation \prec . The pair (X, \prec) is called a *finite partially ordered set* (*finite poset*) if X is finite.

Given a poset (X, \prec) , we define the binary relations $\curvearrowright, \prec^\frown, \simeq_\prec \subseteq X \times X$ as follows:

$$\begin{aligned} a \curvearrowright b &\stackrel{df}{\iff} \neg(a \prec b) \wedge \neg(b \prec a) \wedge a \neq b \\ a \prec^\frown b &\stackrel{df}{\iff} a \prec b \vee a \curvearrowright b \\ a \simeq_\prec b &\stackrel{df}{\iff} a = b \vee a \curvearrowright b \end{aligned}$$

In other words, we write $a \frown_{\prec} b$ if a and b are *distinctly incomparable* elements of X w.r.t. the partial order \prec ; we write $a \prec \cap b$ if a and b are distinct and $\neg(b \prec a)$. The relation \simeq_{\prec} was introduced to make some formulations shorter.

A poset (X, \prec) is

- *total* (or *linear*) iff \frown_{\prec} is empty, i.e., for all $a, b \in X$, either $a \prec b$, or $b \prec a$, or $a = b$,
- *stratified* (or *weak*) iff \simeq_{\prec} is an equivalence relation.

Evidently every total order is stratified.

Let \prec_1 and \prec_2 be partial orders on a set X . Then \prec_2 is an *extension* of \prec_1 if $\prec_1 \subseteq \prec_2$. The relation \prec_2 is a *total extension* (*stratified extension*) of \prec_1 if \prec_2 is total (stratified) and $\prec_1 \subseteq \prec_2$.

For a poset (X, \prec) , we define

$$\begin{aligned} \text{Total}_X(\prec) &\stackrel{\text{df}}{=} \{ \triangleleft \subseteq X \times X \mid \triangleleft \text{ is a total extension of } \prec \} \\ \text{Strat}_X(\prec) &\stackrel{\text{df}}{=} \{ \triangleleft \subseteq X \times X \mid \triangleleft \text{ is a stratified extension of } \prec \} \end{aligned}$$

Theorem 2.1 (Szpilrajn's Theorem [28]). For every poset (X, \prec) , $\prec = \bigcap_{\triangleleft \in \text{Total}_X(\prec)} \triangleleft$. \square

Szpilrajn's Theorem states that every partial order is uniquely determined by the intersection of all of its total extensions. The same is also true for stratified extensions.

Corollary 2.1. For every poset (X, \prec) , $\prec = \bigcap_{\triangleleft \in \text{Strat}_X(\prec)} \triangleleft$.

PROOF. Since for each $\triangleleft \in \text{Strat}_X(\prec)$ we have $\prec \subseteq \triangleleft$, then $\prec \subseteq \bigcap_{\triangleleft \in \text{Strat}_X(\prec)} \triangleleft$. Since each total order is a stratified order, it follows that $\text{Total}_X(\prec) \subseteq \text{Strat}_X(\prec)$. Thus, we have $\bigcap_{\triangleleft \in \text{Strat}_X(\prec)} \triangleleft \subseteq \bigcap_{\triangleleft \in \text{Total}_X(\prec)} \triangleleft = \prec$. \square

2.2. Monoids and Equational Monoids

A triple $(X, *, \mathbb{1})$, where X is a set, $*$ is a total binary operation on X , and $\mathbb{1} \in X$, is called a *monoid*, if $(a * b) * c = a * (b * c)$ and $a * \mathbb{1} = \mathbb{1} * a = a$, for all $a, b, c \in X$.

A nonempty equivalence relation $\sim \subseteq X \times X$ is a *congruence* in the monoid $(X, *, \mathbb{1})$ if for all $a_1, a_2, b_1, b_2 \in X$, $a_1 \sim b_1 \wedge a_2 \sim b_2 \Rightarrow (a_1 * a_2) \sim (b_1 * b_2)$.

The triple $(X / \sim, \otimes, [\mathbb{1}])$, where $[a] \otimes [b] = [a * b]$, is called the *quotient monoid* of $(X, *, \mathbb{1})$ under the congruence \sim . The mapping $\phi : X \rightarrow X / \sim$ defined as $\phi(a) = [a]$ is called the *natural homomorphism* generated by the congruence \sim (for more details see for example [3]). The symbols $*$ and \otimes are often omitted if this does not lead to any discrepancy.

Definition 2.1 (Equation monoid (cf. [16, 25])). Let $M = (X, *, \mathbb{1})$ be a *monoid* and let $EQ = \{ x_i = y_i \mid i = 1, \dots, n \}$ be a finite set of *equations*. Define \equiv_{EQ} to be the *least congruence* on M satisfying, $x_i = y_i \Rightarrow x_i \equiv_{EQ} y_i$, for every equation $x_i = y_i \in EQ$. We call the relation \equiv_{EQ} the *congruence defined by EQ*, or *EQ-congruence*.

The *quotient monoid* $M_{\equiv_{EQ}} = (X / \equiv_{EQ}, \otimes, [\mathbb{1}])$, where $[x] \otimes [y] = [x * y]$, is called an *equational monoid*. \blacksquare

The following folklore result shows that the relation \equiv_{EQ} can also be *uniquely* defined in an explicit way.

Proposition 2.1. *For equational monoids, we have the EQ-congruence $\equiv = (\approx \cup \approx^{-1})^*$, where the relation $\approx \subseteq X \times X$ is defined as:*

$$x \approx y \stackrel{df}{\iff} \exists x_1, x_2 \in X. \exists (u = w) \in EQ. x = x_1 * u * x_2 \wedge y = x_1 * w * x_2.$$

PROOF. Define $\approx = \approx \cup \approx^{-1}$. Clearly $(\approx)^*$ is an equivalence relation. Let $x_1 \equiv y_1$ and $x_2 \equiv y_2$. This means $x_1 (\approx)^k y_1$ and $x_2 (\approx)^l y_2$ for some $k, l \geq 0$. Hence, $x_1 * x_2 (\approx)^k y_1 * x_2 (\approx)^l y_1 * y_2$, i.e., $x_1 * x_2 \equiv y_1 * y_2$. Thus, \equiv is a congruence. Let \sim be a congruence satisfying for all $(u = w) \in EQ$, $u \sim w$. Clearly we have $x \approx y \implies x \sim y$. Hence, $x \equiv y \iff x (\approx)^m y \implies x \sim^m y \implies x \sim y$. Thus, the congruence \equiv is the least. \square

Monoids of traces, comtraces and generalized comtraces are all special cases of equational monoids.

2.3. Sequences, Step Sequences and Partial Orders

By an *alphabet* we shall understand any finite set. For an alphabet Σ , let Σ^* denote the set of all *finite* sequences of elements (words) of Σ , let λ denotes the empty sequence, and any subset of Σ^* is called a *language*. In the scope of this paper, we only deal with *finite* sequences. Let “.” denote the sequence concatenation operator (usually omitted). Since the sequence concatenation operator is associative, the triple $(\Sigma^*, \cdot, \lambda)$ is a *monoid* (of sequences).

Consider an alphabet $\mathbb{S} \subseteq \widehat{\mathcal{P}}(X)$ for some finite X . The elements of \mathbb{S} are called *steps* and the elements of \mathbb{S}^* are called *step sequences*. For example if $\mathbb{S} = \{\{a, b, c\}, \{a, b\}, \{a\}, \{c\}\}$ then $\{a, b\}\{c\}\{a, b, c\} \in \mathbb{S}^*$ is a step sequence. The triple $(\mathbb{S}^*, *, \lambda)$, where “*” denotes the step sequence concatenation operator (usually omitted), is a *monoid* (of step sequences), since the step sequence operator is also associative.

We will now show the formal relationship between step sequences and stratified orders. Let $t = A_1 \dots A_k$ be a step sequence. We define $|t|_a$, the number of occurrences of an event a in w , as $|t|_a \stackrel{df}{=} |\{A_i \mid 1 \leq i \leq k \wedge a \in A_i\}|$, where $|X|$ denotes the cardinality of the set X . Then:

- We can uniquely construct its *enumerated step sequence* \bar{t} as

$$\bar{t} \stackrel{df}{=} \bar{A}_1 \dots \bar{A}_k, \text{ where } \bar{A}_i \stackrel{df}{=} \left\{ e^{(|A_1 \dots A_{i-1}|_e + 1)} \mid e \in A_i \right\}.$$

We will call such $\alpha = e^{(i)} \in \bar{A}_i$ an *event occurrence* of e . For each event occurrence $\alpha = e^{(i)}$, let $l(\alpha)$ denote the *label* of α , i.e., $l(\alpha) = l(e^{(i)}) = e$. For instance, if $u = \{a, b\}\{b, c\}\{c, a\}\{a\}$, then $\bar{u} = \{a^{(1)}, b^{(1)}\}\{b^{(2)}, c^{(1)}\}\{a^{(2)}, c^{(2)}\}\{a^{(3)}\}$. Conversely, from an enumerated step sequence $\bar{t} = \bar{A}_1 \dots \bar{A}_k$, we can also uniquely reconstruct its step sequence $t = l[\bar{A}_1] \dots l[\bar{A}_k]$.

- We let $\Sigma_t = \bigcup_{i=1}^k \bar{A}_i$ denote the set of all event occurrences in all steps of t . For example, when $t = \{a, b\}\{b, c\}\{c, a\}\{a\}$, $\Sigma_t = \{a^{(1)}, a^{(2)}, a^{(3)}, b^{(1)}, b^{(2)}, c^{(1)}, c^{(2)}\}$.
- For each $\alpha \in \Sigma_u$, we let $pos_u(\alpha)$ denote the consecutive number of a step where α belongs, i.e., if $\alpha \in \bar{A}_j$ then $pos_u(\alpha) = j$. For our example, $pos_u(a^{(2)}) = 3$, $pos_u(b^{(2)}) = 2$, etc.

Given a step sequence u , we define two relations $\triangleleft_u, \simeq_u \subseteq \Sigma_u \times \Sigma_u$ as:

$$\alpha \triangleleft_u \beta \stackrel{df}{\iff} pos_u(\alpha) < pos_u(\beta) \quad \text{and} \quad \alpha \simeq_u \beta \stackrel{df}{\iff} pos_u(\alpha) = pos_u(\beta).$$

Since $\triangleleft_u^\wedge = \triangleleft_u \cup \simeq_u$, we have $\alpha \triangleleft_u^\wedge \beta \iff (\alpha \neq \beta \wedge pos_u(\alpha) \leq pos_u(\beta))$. Note that \triangleleft_u is a stratified order iff \simeq_u is an equivalence relation on Σ_u . The two propositions below are known folklore results (which are rarely formally proved). We will provide proofs to make the paper self-sufficient. The first proposition shows that \triangleleft_u is indeed a stratified order.

Proposition 2.2. *Given a step sequence $u = B_1 \dots B_n$, the relation \simeq_u is an equivalence relation.*

PROOF. Since $\alpha \simeq_u \beta \iff pos_u(\alpha) = pos_u(\beta)$, it follows that $\alpha, \beta \in \overline{B_i}$ for some $1 \leq i \leq n$. Hence, \simeq_u is an equivalence relation induced by the partitions $\overline{B_1}, \dots, \overline{B_n}$ of Σ_u . \square

The stratified order \triangleleft_u is an order *generated by the step sequence u* .

Conversely, let \triangleleft be a stratified order on a set Σ . The set Σ can be represented as a sequence of equivalence classes $\Omega_\triangleleft = B_1 \dots B_k$ ($k \geq 0$) such that

$$\triangleleft = \bigcup_{i < j} (B_i \times B_j) \quad \text{and} \quad \simeq_\triangleleft = \bigcup_i (B_i \times B_i).$$

The sequence Ω_\triangleleft is a *step sequence representing \triangleleft* .

The correctness of the existence of Ω_\triangleleft is shown by the next folklore proposition.

Proposition 2.3. *If \triangleleft is a stratified order on a set Σ and A, B are two distinct equivalence classes of \simeq_\triangleleft , then either $A \times B \subseteq \triangleleft$ or $B \times A \subseteq \triangleleft$.*

PROOF. Since both equivalence classes A and B are *non-empty*, we let $a \in A$ and $b \in B$. Clearly, $a \triangleleft b$ or $b \triangleleft a$; otherwise, $a \frown_\triangleleft b$, which contradicts that a, b are elements from two distinct equivalence classes. There are two cases:

1. If $a \triangleleft b$: we want to show $A \times B \subseteq \triangleleft$. Let $c \in A$ and $d \in B$, it suffices to show $c \triangleleft d$. Assume for contradiction that $\neg(c \triangleleft d)$. Since $c \not\triangleleft_\triangleleft d$, it follows that $d \triangleleft c$. There are three subcases:
 - (a) If $a = c$, then $d \triangleleft a$ and $a \triangleleft b$. Hence, $d \triangleleft b$. This contradicts that $b, d \in B$.
 - (b) If $b = d$, then $b \triangleleft c$ and $a \triangleleft b$. Hence, $a \triangleleft c$. This contradicts that $a, c \in A$.
 - (c) If $a \neq c$ and $b \neq d$, then $a \frown_\triangleleft c$ and $b \frown_\triangleleft d$ and $\neg(a \frown_\triangleleft d)$ and $\neg(c \frown_\triangleleft b)$. Since $\neg(a \frown_\triangleleft d)$, either $a \triangleleft d$ or $d \triangleleft a$.
 - If $a \triangleleft d$: since $d \triangleleft c$, it follows $a \triangleleft c$. This contradicts $a \frown_\triangleleft c$.
 - If $d \triangleleft a$: since $a \triangleleft b$, it follows $d \triangleleft b$. This contradicts $d \frown_\triangleleft b$.

Therefore, we conclude $A \times B \subseteq \triangleleft$.

2. If $b \triangleleft a$: using a dual argument to (1), we can show that $B \times A \subseteq \triangleleft$. \square

The idea of Proposition 2.3 is that if we define a relation $\widehat{\triangleleft}$ on the set of equivalence classes $\{B_1, \dots, B_n\}$ of \simeq_\triangleleft such that $B_i \widehat{\triangleleft} B_j \iff B_i \times B_j \subseteq \triangleleft$, then $\widehat{\triangleleft}$ is a total order on $\{B_1, \dots, B_n\}$. Hence, Propositions 2.2 and 2.3 are fundamental for understanding the *equivalence* of stratified partial orders and step sequences.

Note that since sequences are special cases of step sequences (step sequences of singletons) and total orders are special cases of stratified orders, the above results can be applied to sequences and finite total orders as well. Hence, for each sequence $x \in \Sigma^*$, we let \triangleleft_x denote the *total order generated by x* , and for every total order \triangleleft , we let Ω_\triangleleft denote the *sequence generating \triangleleft* .

3. Traces vs. Partial Orders

Traces or partially commutative monoids [2, 5, 23, 24] are *equational monoids over sequences*. In the previous section we have shown how sequences correspond to finite total orders and how step sequences correspond to finite stratified orders. In this section we will discuss the relationship between traces and finite partial orders.

The theory of traces has been utilized to tackle problems from quite diverse areas including combinatorics, graph theory, algebra, logic and, especially (due to the relationship to partial orders) concurrency theory [5, 23, 24].

Since traces constitute a *sequence representation of partial orders*, they can effectively model “true concurrency” in various aspects of concurrency theory using relatively simple and intuitive means. We will now recall the definition of a *trace monoid*.

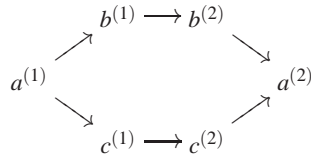
Definition 3.1 ([5, 24]). Let $M = (E^*, *, \lambda)$ be a *free monoid* generated by E , and let the relation $ind \subseteq E \times E$ be an irreflexive and symmetric relation (called *independency* or *commutation*), and $EQ \stackrel{df}{=} \{ab = ba \mid (a, b) \in ind\}$. Let \equiv_{ind} , called *trace congruence*, be the congruence defined by EQ . Then the equational monoid $M_{\equiv_{ind}} = (E^* / \equiv_{ind}, \otimes, [\lambda])$ is a monoid of *traces* (or a *free partially commutative monoid*). The pair (E, ind) is called a *trace alphabet*. ■

We will omit the subscript ind from trace congruence and write \equiv if it causes no ambiguity.

Example 3.1. Let $E = \{a, b, c\}$, $ind = \{(b, c), (c, b)\}$, i.e., $EQ = \{bc = cb\}$. For example, $abcbca \equiv accbba$ (since $abcbca \approx acbbca \approx acbcbca \approx accbba$). Also we have $\mathbf{t}_1 = [abcbca] = \{abcbca, abccba, acbbca, acbcbca, abbcca, accbba\}$, $\mathbf{t}_2 = [abc] = \{abc, acb\}$ and $\mathbf{t}_3 = [bca] = \{bca, cba\}$ are traces. Note that $\mathbf{t}_1 = \mathbf{t}_2 \otimes \mathbf{t}_3$ since $[abcbca] = [abc] \otimes [bca]$. ■

Each trace can be interpreted as a finite partial order. Let $\mathbf{t} = \{x_1, \dots, x_k\}$ be a trace, and let \triangleleft_{x_i} denotes the total order induced by the sequence x_i , $i = 1, \dots, k$. Note that $\Sigma_{x_i} = \Sigma_{x_j}$ for all $i, j = 1, \dots, n$, so we can define $\Sigma_{\mathbf{t}} = \Sigma_{x_i}$, $i = 1, \dots, n$. For example for \mathbf{t}_1 from Example 3.1 we have $\Sigma_{\mathbf{t}_1} = \{a^{(1)}, b^{(1)}, c^{(1)}, a^{(2)}, b^{(2)}, c^{(2)}\}$. Clearly $\triangleleft_i \subseteq \Sigma_{\mathbf{t}} \times \Sigma_{\mathbf{t}}$. The partial order generated by \mathbf{t} can then be defined as $\prec_{\mathbf{t}} = \bigcap_{i=1}^k \triangleleft_{x_i}$. In fact, the set $\{\triangleleft_{x_1}, \dots, \triangleleft_{x_k}\}$ consists of all the total extensions of $\prec_{\mathbf{t}}$ (see [23, 24]).

For example, the trace $\mathbf{t}_1 = [abcbca]$ from Example 3.1 can be interpreted as a partial order $\prec_{\mathbf{t}_1}$ depicted in the following diagram (arcs inferred from transitivity are omitted for simplicity):



Remark 3.1. Given a sequence s , to construct the partial order $\prec_{[s]}$ generated by $[s]$, we *do not* need to build up to exponentially many elements of $[s]$. We can simply construct the direct acyclic graph $(\Sigma_{[s]}, \prec_s)$, where $x^{(i)} \prec_s y^{(j)}$ iff $x^{(i)}$ occurs before $y^{(j)}$ on the sequence s and $(x, y) \notin ind$. The relation \prec_s is usually *not* the same as the partial order $\prec_{[s]}$. However, after applying the *transitive closure* operator, we have $\prec_{[s]} = \prec_s^+$ (cf. [5]). We will later see how this idea is generalized to the construction of so-structures and gso-structures from their “trace” representations. Note that

to do so, it is inevitable that we have to generalize the *transitive closure* operator to these order structures. ■

Conversely, each finite partial order can be represented by a trace as follows. Given a finite set X , let (X, \prec) be a poset and let $\{\prec_1, \dots, \prec_k\}$ be the set of all total extensions of \prec . Let $x_i \in X^*$ be a sequence that represents \prec_i , for $i = 1, \dots, k$. Then the set $\mathbf{t}_\prec = \{x_1, \dots, x_k\}$ is a trace over the trace alphabet (X, \curvearrowright) , i.e., $\mathbf{t}_\prec = [x_i] / \equiv_{\curvearrowright}$ for any $i = 1, \dots, k$.

From the concurrency point of view, the fundamental advantage of traces is that in many cases it is simpler and more fruitful to analyze sequences equipped with a independency relation *ind* and their underlying quotient trace monoid than their equivalent partial order representations. This is especially the case when we want to study the formal linguistic aspects of concurrent behaviors, e.g., Ochmanski’s characterization of recognizable trace language [25] and Zielonka’s theory of asynchronous automata [30]. For more details on traces and their various properties, the reader is referred to the monograph [5].

4. Relational Structures Model of Concurrency

Even though partial orders are a principle tool for modelling “true concurrency,” they have some limitations. While they can sufficiently model the “earlier than” relationship, they cannot model neither the “not later than” relationship nor the “non-simultaneously” relationship. It was shown in [11] that any reasonable concurrent behavior can be modelled by an appropriate *pair of relations*. This leads to the theory of *relational structures models of concurrency* [7, 9, 13] (see [9] for a detailed bibliography and history).

In this chapter, we review the theory of *stratified order structures* of [13] and *generalized stratified order structures* of [7, 9]. The former can model both the “earlier than” and the “not later than” relationships, but not the “non-simultaneously” relationship. The latter can model all three relationships.

While traces provide sequence representations of causal partial orders, their extensions, com-traces and generalized comtraces discussed in the following sections, are *step sequence* representations of stratified order structures and generalized stratified order structures respectively.

Since the theory of relational order structures is far less known than the theory of causal partial orders, we will not only give appropriate definitions but also introduce some intuition and motivation behind those definitions using simple examples.

We start with the concept of an *observation*:

An *observation* (also called a *run* or an *instance of concurrent behavior*) is an abstract model of the execution of a concurrent system.

It was argued in [11] that *an observation must be a total, stratified or interval order* (interval orders are not used in this paper). Totally ordered observations can be represented by sequences while stratified observations can be represented by step sequences.

The next concept is a *concurrent behavior*:

A *concurrent behavior* (*concurrent history*) is a set of equivalent observations.

When totally ordered observations are sufficient to define whole concurrent behaviors, then the concurrent behaviors can entirely be described by causal partial orders. However if sophisticated sets of stratified observations are used to describe concurrent behaviors, e.g., to model the “not later than” relationship, we need to use appropriate relational structures [11].

4.1. Stratified Order Structure

By a *relational structure*, we will mean a triple $T = (X, R_1, R_2)$, where X is a set and R_1, R_2 are binary relations on X . A relational structure $T' = (X', R'_1, R'_2)$ is an *extension* of T , denoted as $T \subseteq T'$, iff $X = X'$, $R_1 \subseteq R'_1$ and $R_2 \subseteq R'_2$.

Definition 4.1 (Stratified order structure [13]). A *stratified order structure (so-structure)* is a relational structure $S = (X, \prec, \sqsubseteq)$, such that for all $a, b, c \in X$, the following hold:

$$\begin{array}{ll} \text{S1:} & a \not\sqsubseteq a \\ \text{S2:} & a \prec b \implies a \sqsubseteq b \\ \text{S3:} & a \sqsubseteq b \sqsubseteq c \wedge a \neq c \implies a \sqsubseteq c \\ \text{S4:} & a \sqsubseteq b \prec c \vee a \prec b \sqsubseteq c \implies a \prec c \end{array}$$

When X is finite, S is called a *finite so-structure*. ■

Note that the axioms S1–S4 imply that (X, \prec) is a poset and $a \prec b \implies b \not\sqsubseteq a$. The relation \prec is called *causality* and represents the “earlier than” relationship, and the relation \sqsubseteq is called *weak causality* and represents the “not later than” relationship. The axioms C1–C4 model the mutual relationship between “earlier than” and “not later than” relations, *provided that the system runs are defined as stratified orders*.

The concept of so-structures were independently introduced in [6] and [10] (the axioms are slightly different from S1–S4, although equivalent). Their comprehensive theory has been presented in [13]. They have been successfully applied to model inhibitor and priority systems, asynchronous races, synthesis problems, etc. (see for example [12, 18, 17, 19, 20, 26] and others).

The name partially follows from the following result.

Proposition 4.1 ([11]). For every stratified order \triangleleft on X , the triple $S_{\triangleleft} = (X, \triangleleft, \triangleleft^{\cap})$ is a so-structure. □

Definition 4.2 (Stratified extension of a so-structure [13]). Let $S = (X, \prec, \sqsubseteq)$ be a so-structure. A stratified order \triangleleft on X is a *stratified extension* of S if for all $\alpha, \beta \in X$,

$$\alpha \prec \beta \implies \alpha \triangleleft \beta \quad \text{and} \quad \alpha \sqsubseteq \beta \implies \alpha \triangleleft^{\cap} \beta$$

The set of all stratified extensions of S is denoted as $\text{ext}(S)$. ■

According to Szpilrajn’s Theorem, every poset can be reconstructed by taking the intersection of all of its total extensions. A similar result holds for so-structures and stratified extensions.

Theorem 4.1 ([13, Theorem 2.9]). Let $S = (X, \prec, \sqsubseteq)$ be a so-structure. Then

$$S = \left(X, \bigcap_{\triangleleft \in \text{ext}(S)} \triangleleft, \bigcap_{\triangleleft \in \text{ext}(S)} \triangleleft^{\cap} \right).$$

□

The set $ext(S)$ also has the following internal property that will be useful in various proofs.

Theorem 4.2 ([11]). *Let $S = (X, \prec, \sqsubset)$ be a so-structure. Then for every $a, b \in X$*

$$\left((\exists \triangleleft \in ext(S). a \triangleleft b) \wedge (\exists \triangleleft \in ext(S). b \triangleleft a) \right) \implies (\exists \triangleleft \in ext(S). a \triangleleft^\frown b). \quad \square$$

The classification of concurrent behaviors provided in [11] says that a concurrent behavior conforms to the paradigm⁴ π_3 if it has the same property as stated in Theorem 4.2 for $ext(S)$. In other words, Theorem 4.2 states that the set $ext(S)$ conforms to paradigm π_3 of [11].

4.2. Generalized Stratified Order Structure

The stratified order structures can adequately model concurrent histories only when paradigm π_3 is satisfied. For the general case, we need *gso-structures* introduced in [7] also under the assumption that the system runs are defined as stratified orders.

Definition 4.3 (Generalized stratified order structure [7, 9]). A *generalized stratified order structure* (*gso-structure*) is a relational structure $G = (X, \diamond, \sqsubset)$ such that \sqsubset is irreflexive, \diamond is symmetric and irreflexive, and the triple $S_G = (X, \prec_G, \sqsubset)$, where $\prec_G = \diamond \cap \sqsubset$, is a so-structure.

Such relational structure S_G is called the *so-structure induced by G* . When X is finite, G is called a *finite gso-structure*. ■

The relation \diamond is called *commutativity* and represents the “non-simultaneously” relationship, while the relation \sqsubset is called *weak causality* and represents the “not later than” relationship.

For a binary relation R on X , we let R^{sym} denote the *symmetric closure* of R and R^{sym} can be defined as $R^{\text{sym}} \stackrel{\text{df}}{=} R \cup R^{-1}$.

Definition 4.4 (Stratified extension of a gso-structure [7, 9]). Let $G = (X, \diamond, \sqsubset)$ be a gso-structure. A stratified order \triangleleft on X is a *stratified extension* of G if for all $\alpha, \beta \in X$,

$$\alpha \diamond \beta \implies \alpha \triangleleft^{\text{sym}} \beta \quad \text{and} \quad \alpha \sqsubset \beta \implies \alpha \triangleleft^\frown \beta$$

The set of all stratified extensions of G is denoted as $ext(G)$. ■

Every gso-structure can also be uniquely reconstructed from its stratified extensions. The generalization of Szpilrajn’s Theorem for gso-structures can be stated as following.

Theorem 4.3 ([7, 9]). *Let $G = (X, \diamond, \sqsubset)$ be a gso-structure. Then*

$$G = \left(X, \bigcap_{\triangleleft \in ext(G)} \triangleleft^{\text{sym}}, \bigcap_{\triangleleft \in ext(G)} \triangleleft^\frown \right). \quad \square$$

The gso-structures *do not* have an equivalence of Theorem 4.2, which makes proving properties about gso-structures more difficult, but they can model the most general concurrent behaviors (provided that observations are stratified orders) [9].

⁴A *paradigm* is a supposition or statement about the structure of a concurrent behavior (concurrent history) involving a treatment of *simultaneity*. See [9, 11] for more details.

4.3. Motivating Example

To understand the main motivation and intuition behind the use of so-structures and gso-structures, we will consider four simple programs in the following example (from [9]).

Example 4.1. All the programs are written using a mixture of *cobegin*, *coend* and a version of *concurrent guarded commands*.

```

P1:  begin int x,y;
      a:  begin x:=0; y:=0 end;
      cobegin b:  x:=x+1, c:  y:=y+1 coend
    end P1.

P2:  begin int x,y;
      a:  begin x:=0; y:=0 end;
      cobegin b:  x=0 → y:=y+1, c:  x:=x+1 coend
    end P2.

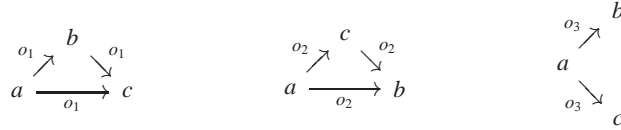
P3:  begin int x,y;
      a:  begin x:=0; y:=0 end;
      cobegin b:  y=0 → x:=x+1, c:  x=0 → y:=y+1 coend
    end P3.

P4:  begin int x;
      a:  x:=0;
      cobegin b:  x:=x+1, c:  x:=x+2 coend
    end P4.

```

Each program is a different composition of three events (actions) called a , b , and c (a_i , b_i , c_i , $i = 1, \dots, 4$, to be exact, but a restriction to a , b , c does not change the validity of the analysis below, while simplifying the notation). Transition systems modelling these programs are shown in Figure 1. ■

Let $obs(P_i)$ denote the set of all program runs involving the actions a, b, c that can be observed. Assume that simultaneous executions can be observed. In this simple case all runs (or observations) can be modelled by *step sequences*. Let us denote $o_1 = \{a\}\{b\}\{c\}$, $o_2 = \{a\}\{c\}\{b\}$, $o_3 = \{a\}\{b, c\}$. Each o_i can be equivalently seen as a stratified partial order $o_i = (\{a, b, c\}, \xrightarrow{o_i})$ where:



We can now write $obs(P_1) = \{o_1, o_2, o_3\}$, $obs(P_2) = \{o_1, o_3\}$, $obs(P_3) = \{o_3\}$, $obs(P_4) = \{o_1, o_2\}$. Note that for every $i = 1, \dots, 4$, all runs from the set $obs(P_i)$ yield exactly the same outcome. Hence, each $obs(P_i)$ is called the *concurrent history* of P_i .

An abstract model of such an outcome is called a *concurrent behavior*, and now we will discuss how causality, weak causality and commutativity relations are used to construct concurrent behavior.

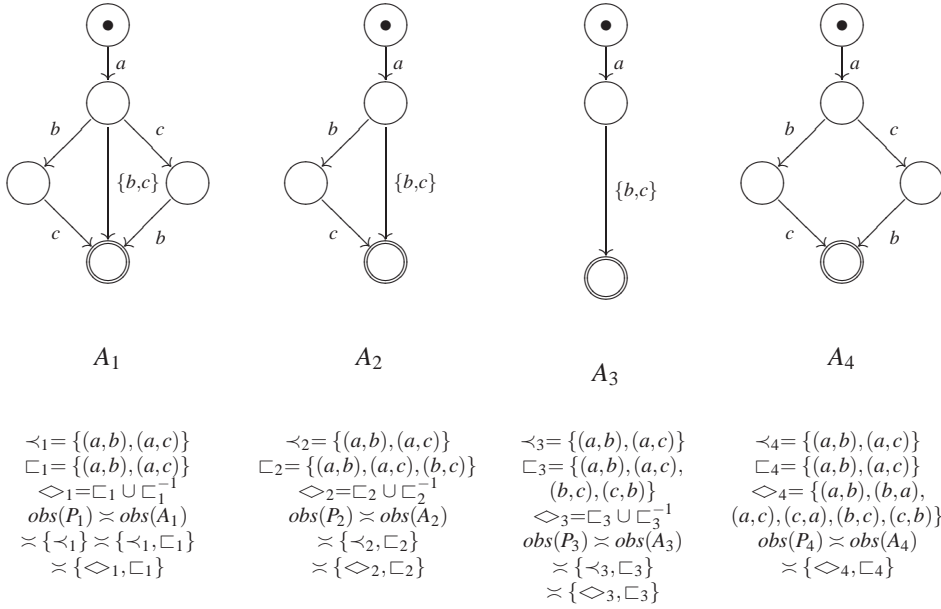


Figure 1: Examples of *causality*, *weak causality*, and *commutativity*. Each program P_i can be modelled by a labelled transition system (automaton) A_i . The step $\{a,b\}$ denotes the *simultaneous* execution of a and b .

Program P_1 :

In the set $obs(P_1)$, for each run, a always precedes both b and c , and there is no *causal* relationship between b and c . This *causality* relation, \prec , is the partial order defined as $\prec = \{(a,b), (a,c)\}$. In general \prec is defined by: $x \prec y$ iff for each run o we have $x \xrightarrow{o} y$. Hence for P_1 , \prec is the intersection of o_1 , o_2 and o_3 , and $\{o_1, o_2, o_3\}$ is the set of all stratified extensions of the relation \prec .

Thus, in this case, the causality relation \prec models the concurrent behavior corresponding to the set of (equivalent) runs $obs(P_1)$. We will say that $obs(P_1)$ and \prec are *tantamount*⁵ and write $obs(P_1) \asymp \{\prec\}$ or $obs(P_1) \asymp (\{a,b,c\}, \prec)$. Having $obs(P_1)$ one may construct \prec (as an intersection), and hence construct $obs(P_4)$ (as the set of all stratified extensions). This is a classical case of the “true” concurrency approach, where concurrent behavior is modelled by a causality relation.

Before considering the remaining cases, note that the causality relation \prec is exactly the same in all four cases, i.e., $\prec_i = \{(a,b), (a,c)\}$, for $i = 1, \dots, 4$, so we may omit the index i .

⁵Following [9] we are using the word “tantamount” instead of “equivalent” as the latter usually implies that the entities are of the same type, as “equivalent automata”, “equivalent expressions”, etc. Tantamount entities can be of different types.

Programs P_2 and P_3 :

To deal with $obs(P_2)$ and $obs(P_3)$, \prec is insufficient because $o_2 \notin obs(P_2)$ and $o_1, o_2 \notin obs(P_2)$. Thus, we need a weak causality relation \sqsubset defined in this context as $x \sqsubset y$ iff for each run o we have $\neg(y \xrightarrow{o} x)$ (x is never executed after y). For our four cases we have $\sqsubset_2 = \{(a, b), (a, c), (b, c)\}$, $\sqsubset_1 = \sqsubset_4 = \prec$, and $\sqsubset_3 = \{(a, b), (a, c), (b, c), (c, b)\}$. Notice again that for $i = 2, 3$, the pair of relations $\{\prec, \sqsubset_i\}$ and the set $obs(P_i)$ are equivalent in the sense that each is definable from the other. (The set $obs(P_i)$ can be defined as the greatest set PO of partial orders built from a, b and c satisfying $x \prec y \Rightarrow \forall o \in PO. x \xrightarrow{o} y$ and $x \sqsubset_i y \Rightarrow \forall o \in PO. \neg(y \xrightarrow{o} x)$.)

Hence again in these cases ($i = 2, 3$) $obs(P_i)$ and $\{\prec, \sqsubset_i\}$ are *tantamount*, $obs(P_i) \asymp \{\prec, \sqsubset_i\}$, and so the pair $\{\prec, \sqsubset_i\}$, $i = 2, 3$, models the concurrent behavior described by $obs(P_i)$. Note that \sqsubset_i alone is not sufficient, since (for instance) $obs(P_2)$ and $obs(P_2) \cup \{\{a, b, c\}\}$ define the same relation \sqsubset .

Program P_4 :

The causality relation \prec does not model the concurrent behavior of P_4 correctly⁶ since o_3 does not belong to $obs(P_4)$. The commutativity relation \diamond is defined in this context as $x \diamond y$ iff for each run o either $x \xrightarrow{o} y$ or $y \xrightarrow{o} x$. For the set $obs(P_4)$, the relation \diamond_4 looks like $\diamond_4 = \{(a, b), (b, a), (a, c), (c, a), (b, c), (c, b)\}$. The pair of relations $\{\diamond_4, \prec\}$ and the set $obs(P_4)$ are equivalent in the sense that each is definable from the other. (The set $obs(P_4)$ is the greatest set PO of partial orders built from a, b and c satisfying $x \diamond_4 y \Rightarrow \forall o \in PO. x \xrightarrow{o} y \vee y \xrightarrow{o} x$ and $x \prec y \Rightarrow \forall o \in PO. x \xrightarrow{o} y$.) In other words, $obs(P_4)$ and $\{\diamond_4, \prec\}$ are *tantamount*, $obs(P_4) \asymp \{\prec, \diamond_4, \prec\}$, so we may say that in this case the relations $\{\diamond_4, \prec\}$ model the concurrent behavior described by $obs(P_4)$.

Note also that $\diamond_1 = \prec \cup \prec^{-1}$ and the pair $\{\diamond_1, \prec\}$ also models the concurrent behavior described by $obs(P_1)$.

The state transition model A_i of each P_i and their respective concurrent histories and concurrent behaviors are summarized in Figure 1. Thus, we can make the following observations:

1. $obs(P_1)$ can be modelled by the relation \prec alone, and $obs(P_1) \asymp \{\prec\}$.
2. $obs(P_i)$, for $i = 1, 2, 3$ can also be modelled by appropriate pairs of relations $\{\prec, \sqsubset_i\}$, and $obs(P_i) \asymp \{\prec, \sqsubset_i\}$.
3. all sets of observations $obs(P_i)$, for $i = 1, 2, 3, 4$ are modelled by appropriate pairs of relations $\{\diamond_i, \sqsubset_i\}$, and $obs(P_i) \asymp \{\diamond_i, \sqsubset_i\}$.

Note that the relations $\prec, \diamond, \sqsubset$ are not independent, since it can be proved (see [11]) that $\prec = \diamond \cap \sqsubset$. Intuitively, since \diamond and \sqsubset are the abstraction of the “earlier than or later than” and “not later than” relations, it follows that their intersection is the abstraction of the “earlier than” relation. \square

5. Comtraces

The standard definition of a free monoid $(E^*, *, \lambda)$ assumes the elements of E have no internal structure (or their internal structure does not affect any monoidal properties), and they are

⁶ Unless we assume that simultaneity is not allowed, or not observed, in which case $obs(P_1) = obs(P_4) = \{o_1, o_2\}$, $obs(P_2) = \{o_1\}$, $obs(P_3) = \emptyset$.

often called ‘letters’, ‘symbols’, ‘names’, etc. When we assume the elements of E have some internal structure, for instance that they are sets, this internal structure may be used when defining the set of equations EQ . This idea is exploited in the concept of a *comtrace*.

Comtraces (combined traces), were introduced in [12] as an extension of traces to distinguish between “earlier than” and “not later than” phenomena, are equational monoids of step sequence monoids. The equations EQ are in this case defined implicitly via two relations *simultaneity* and *serializability*.

Definition 5.1 (Comtrace alphabet [12]). Let E be a finite set (of events) and let $ser \subseteq sim \subseteq E \times E$ be two relations called *serializability* and *simultaneity* respectively and the relation sim is irreflexive and symmetric. Then the triple (E, sim, ser) is called the *comtrace alphabet*. ■

Intuitively, if $(a, b) \in sim$ then a and b can occur simultaneously (or be a part of a *synchronous* occurrence in the sense of [18]), while $(a, b) \in ser$ means that a and b may occur simultaneously or a may occur before b (i.e., both executions are equivalent). We define \mathbb{S} , the set of all (potential) *steps*, as the set of all cliques of the graph (E, sim) , i.e.,

$$\mathbb{S} \stackrel{df}{=} \{A \mid A \neq \emptyset \wedge \forall a, b \in A. (a = b \vee (a, b) \in sim)\}.$$

Definition 5.2 (Comtrace congruence [12]). Let $\theta = (E, sim, ser)$ be a comtrace alphabet and let \equiv_{ser} , called *comtrace congruence*, be the EQ -congruence defined by the set of equations

$$EQ \stackrel{df}{=} \{A = BC \mid A = B \cup C \in \mathbb{S} \wedge B \times C \subseteq ser\}.$$

Then the equational monoid $(\mathbb{S}^* / \equiv_{ser}, \otimes, [\lambda])$ is called a monoid of *comtraces* over θ . ■

Since ser is irreflexive, for each $(A = BC) \in EQ$ we have $B \cap C = \emptyset$. By Proposition 2.1, the comtrace congruence relation can also be defined explicitly in non-equational form as follows.

Proposition 5.1. Let $\theta = (E, sim, ser)$ be a comtrace alphabet and let \mathbb{S}^* be the set of all step sequences defined on θ . Let $\approx_{ser} \subseteq \mathbb{S}^* \times \mathbb{S}^*$ be the relation comprising all pairs (t, u) of step sequences such that $t = wAz$ and $u = wBcz$, where $w, z \in \mathbb{S}^*$ and A, B, C are steps satisfying $B \cup C = A$ and $B \times C \subseteq ser$. Then $\equiv_{ser} = (\approx_{ser} \cup \approx_{ser}^{-1})^*$. □

We will omit the subscript ser from comtrace congruence and \approx_{ser} , and only write \equiv and \approx if it causes no ambiguity.

Example 5.1. Let $E = \{a, b, c\}$ where a, b and c are three atomic operations, where

$$a: y \leftarrow x + y \qquad b: x \leftarrow y + 2 \qquad c: y \leftarrow y + 1$$

Assuming simultaneous reading is allowed, then only b and c can be performed simultaneously, and the simultaneous execution of b and c gives the same outcome as executing b followed by c (see also the program P_2 of Example 4.1). We can then define $sim = \{(b, c), (c, b)\}$ and $ser = \{(b, c)\}$, and we have $\mathbb{S} = \{\{a\}, \{b\}, \{c\}, \{b, c\}\}$, $EQ = \{\{b, c\} = \{b\}\{c\}\}$. For example, $\mathbf{x} = [\{a\}\{b, c\}] = \{\{a\}\{b, c\}, \{a\}\{b\}\{c\}\}$ is a comtrace. Note that $\{a\}\{c\}\{b\} \notin \mathbf{x}$. ■

Even though traces are quotient monoids over sequences and comtraces are quotient monoids over step sequences (and the fact that steps are sets is used in the definition of quotient congruence), traces can be regarded as a special case of comtraces. In principle, each trace commutativity equation $ab = ba$ corresponds to two comtrace equations $\{a, b\} = \{a\}\{b\}$ and $\{a, b\} = \{b\}\{a\}$. This relationship can formally be formulated as follows.

Let (E, ind) and (E, sim, ser) be trace and comtrace alphabets respectively. For each sequence $x = a_1 \dots a_n \in E^*$, we define $x^{\{\}} = \{a_1\} \dots \{a_n\}$ to be its corresponded sequence of singleton sets.

Lemma 5.1 (Relationship between traces and comtraces).

1. If $ser = sim$, then for each comtrace $\mathbf{t} \in \mathbb{S}^* / \equiv_{ser}$ there is a step sequence $x = \{a_1\} \dots \{a_k\} \in \mathbb{S}^*$ such that $\mathbf{t} = [x]_{\equiv_{ser}}$.
2. If $ser = sim = ind$, then for each $x, y \in E^*$, we have $x \equiv_{ind} y \iff x^{\{\}} \equiv_{ser} y^{\{\}}$.

PROOF. 1. Let $\mathbf{t} = [A_1 \dots A_m]$. Then $\mathbf{t} = [A_1] \dots [A_m]$. Let $A_i = \{a_1^i, \dots, a_{n_i}^i\}$. Then since $ser = sim$, we have $[A_i] = [\{a_1^i\}] \dots [\{a_{n_i}^i\}]$, for $i = 1, \dots, m$.

2. It suffices to show that $x \approx_{ind} y \iff x^{\{\}} (\approx_{ser} \circ \approx_{ser}^{-1}) y^{\{\}}$. Note that if $sim = ser$ then ser is symmetric. We have

$$\begin{aligned}
& x \approx_{ind} y \\
& \iff x = wabz \wedge y = wba z \wedge (a, b) \in ind && \langle \text{Definition of } \equiv_{ind} \rangle \\
& \iff x^{\{\}} = w^{\{\}} \{a\} \{b\} z^{\{\}} \wedge y^{\{\}} = w^{\{\}} \{b\} \{a\} z^{\{\}} \\
& \quad \wedge \{a\} \times \{b\} \subseteq ser && \langle \text{Since } ser = sim = ind \rangle \\
& \iff x^{\{\}} \approx_{ser} w^{\{\}} \{a, b\} z^{\{\}} \wedge w^{\{\}} \{a, b\} z^{\{\}} \approx_{ser}^{-1} y^{\{\}} && \langle \text{Definition of } \equiv_{ser} \rangle
\end{aligned}$$

□

Let \mathbf{t} be a trace over (E, ind) and let \mathbf{v} be a comtrace over (E, sim, ser) .

- We say that \mathbf{t} and \mathbf{v} are *equivalent* if $sim = ser = ind$ and there is $x \in E^*$ such that $\mathbf{t} = [x]_{\equiv_{ind}}$ and $\mathbf{v} = [x^{\{\}}]_{\equiv_{ser}}$. If a trace \mathbf{t} and a comtrace \mathbf{v} are equivalent we will write $\mathbf{t} \stackrel{t \rightsquigarrow c}{\equiv} \mathbf{v}$.

Note that Lemma 5.1 guarantees that this definition is correct.

Proposition 5.2. Let \mathbf{t}, \mathbf{r} be traces and \mathbf{v}, \mathbf{w} be comtraces. Then

1. $\mathbf{t} \stackrel{t \rightsquigarrow c}{\equiv} \mathbf{v} \wedge \mathbf{t} \stackrel{t \rightsquigarrow c}{\equiv} \mathbf{w} \implies \mathbf{v} = \mathbf{w}$.
2. $\mathbf{t} \stackrel{t \rightsquigarrow c}{\equiv} \mathbf{v} \wedge \mathbf{r} \stackrel{t \rightsquigarrow c}{\equiv} \mathbf{v} \implies \mathbf{t} = \mathbf{r}$.

□

Equivalent traces and comtraces generate identical partial orders. However, we will postpone the discussion of this issue to Section 10. Hence *traces can be regarded as a special case of comtraces*.

It appears that the concept of the comtrace can be very useful to formalize the concept of *synchrony* (cf. [18]). In principle events a_1, \dots, a_k are *synchronous* if they can be executed in one step $\{a_1, \dots, a_k\}$ but this execution cannot be modelled by any sequence of proper subsets of $\{a_1, \dots, a_k\}$. Note that in general ‘synchrony’ is not necessarily ‘simultaneity’ as it does not include the concept of time [16]. It appears, however, that the mathematics used to deal with synchrony are very close to that used to deal with simultaneity.

Definition 5.3 (Independency and synchrony). Let $(E, \text{sim}, \text{ser})$ be a given comtrace alphabet. We define the relations ind , syn and the set \mathbb{S}_{syn} as follows:

- $\text{ind} \subseteq E \times E$, called *independency*, and defined as $\text{ind} = \text{ser} \cap \text{ser}^{-1}$,

- $\text{syn} \subseteq E \times E$, called *synchrony*, and defined as:

$$(a, b) \in \text{syn} \stackrel{\text{df}}{\iff} (a, b) \in \text{sim} \setminus \text{ser}^{\text{sym}},$$

- $\mathbb{S}_{\text{syn}} \subseteq \mathbb{S}$, called *synchronous steps*, and defined as:

$$A \in \mathbb{S}_{\text{syn}} \stackrel{\text{df}}{\iff} A \neq \emptyset \wedge (\forall a, b \in A. (a, b) \in \text{syn}). \quad \blacksquare$$

If $(a, b) \in \text{ind}$ then a and b are *independent*, i.e., executing them either simultaneously, or a followed by b , or b followed by a , will yield exactly the same result. If $(a, b) \in \text{syn}$ then a and b are *synchronous*, which means they might be executed in one step, either $\{a, b\}$ or as a part of bigger step, but such an execution is not equivalent to either a followed by b , or b followed by a . In principle, the relation syn is a counterpart of ‘synchrony’ (cf. [18]). If $A \in \mathbb{S}_{\text{syn}}$, then the set of events A can be executed as one step, but it *cannot* be simulated by any sequence of its subsets.

Example 5.2. Assume we have $E = \{a, b, c, d, e\}$, $\text{sim} = \{(a, b), (b, a), (a, c), (c, a), (a, d), (d, a)\}$, and $\text{ser} = \{(a, b), (b, a), (a, c)\}$. Hence, $\mathbb{S} = \{\{a, b\}, \{a, c\}, \{a, d\}, \{a\}, \{b\}, \{c\}, \{e\}\}$, and

$$\text{ind} = \{(a, b), (b, a)\} \quad \text{syn} = \{(a, d), (d, a)\} \quad \mathbb{S}_{\text{syn}} = \{\{a, d\}\}$$

Since $\{a, d\} \in \mathbb{S}_{\text{syn}}$ the step $\{a, d\}$ *cannot* be splitted. For example the comtraces $\mathbf{x}_1 = [\{a, b\}\{c\}\{a\}]$, $\mathbf{x}_2 = [\{e\}\{a, d\}\{a, c\}]$, and $\mathbf{x}_3 = [\{a, b\}\{c\}\{a\}\{e\}\{a, d\}\{a, c\}]$ are the following sets of step sequences:

$$\begin{aligned} \mathbf{x}_1 &= \{\{a, b\}\{c\}\{a\}, \{a\}\{b\}\{c\}\{a\}, \{b\}\{a\}\{c\}\{a\}, \{b\}\{a, c\}\{a\}\} \\ \mathbf{x}_2 &= \{\{e\}\{a, d\}\{a, c\}, \{e\}\{a, d\}\{a\}\{c\}\} \\ \mathbf{x}_3 &= \{\{a, b\}\{c\}\{a\}\{e\}\{a, d\}\{a, c\}, \{a\}\{b\}\{c\}\{a\}\{e\}\{a, d\}\{a, c\}, \\ &\quad \{b\}\{a\}\{c\}\{a\}\{e\}\{a, d\}\{a, c\}, \{b\}\{a, c\}\{a\}\{e\}\{a, d\}\{a, c\}, \\ &\quad \{a, b\}\{c\}\{a\}\{e\}\{a, d\}\{a\}\{c\}, \{a\}\{b\}\{c\}\{a\}\{e\}\{a, d\}\{a\}\{c\}, \\ &\quad \{b\}\{a\}\{c\}\{a\}\{e\}\{a, d\}\{a\}\{c\}, \{b\}\{a, c\}\{a\}\{e\}\{a, d\}\{a\}\{c\}\} \end{aligned}$$

We also have $\mathbf{x}_3 = \mathbf{x}_1 \otimes \mathbf{x}_2$. Note that since $(c, a) \notin \text{ser}$, $\{a, c\} \equiv_{\text{ser}} \{a\}\{c\} \not\equiv_{\text{ser}} \{c\}\{a\}$. \blacksquare

6. Generalized Comtraces

There are reasonable concurrent behaviors that cannot be modelled by any comtrace. Let us analyze the following example.

Example 6.1. Let $E = \{a, b, c\}$ where a , b and c are three atomic operations defined as follows (we assume simultaneous reading is allowed):

$$a : x \leftarrow x + 1 \quad b : x \leftarrow x + 2 \quad c : y \leftarrow y + 1$$

It is reasonable to consider them all as ‘concurrent’ as any order of their executions yields exactly the same results (see [11, 13] for more motivation and formal considerations as well as the program P_4 of Example 4.1). Note that while simultaneous execution of $\{a, c\}$ and $\{b, c\}$ are allowed, the step $\{a, b\}$ is *not*, since simultaneous writing on the same variable x is not allowed!

The set of all equivalent executions (or runs) involving one occurrence of the operations a , b and c , and modelling the above case,

$$\mathbf{x} = \{\{a\}\{b\}\{c\}, \{a\}\{c\}\{b\}, \{b\}\{a\}\{c\}, \{b\}\{c\}\{a\}, \{c\}\{a\}\{b\}, \{c\}\{b\}\{a\}, \\ \{a, c\}\{b\}, \{b, c\}\{a\}, \{b\}\{a, c\}, \{a\}\{b, c\}\},$$

is a valid concurrent history or behavior [11, 13]. However x is *not* a comtrace. The problem is that we have $\{a\}\{b\} \equiv \{b\}\{a\}$ but $\{a, b\}$ is *not* a valid step, so no comtrace can represent this situation. ■

In this section, we will introduce the concept of *generalized comtraces* (*g-comtraces*), an extension of comtraces, also equational monoids of step sequences. The g-comtraces will be able to model “non-simultaneously” relationship similar to the one from Example 6.1.

Definition 6.1 (Generalized comtrace alphabet). Let E be a finite set (of events). Let ser , sim and inl be three relations on E called *serializability*, *simultaneity* and *interleaving* respectively satisfying the following conditions:

- sim and inl are irreflexive and symmetric,
- $ser \subseteq sim$, and
- $sim \cap inl = \emptyset$.

Then the triple (E, sim, ser, inl) is called a *g-comtrace alphabet*. ■

The interpretation of the relations sim and ser is as in Definition 5.1, and $(a, b) \in inl$ means a and b cannot occur simultaneously, but their occurrence in any order is equivalent. As for comtraces, we define \mathbb{S} , the set of all (potential) *steps*, as the set of all cliques of the graph (E, sim) .

Definition 6.2 (Generalized comtrace congruence). Let $\Theta = (E, sim, ser, inl)$ be a g-comtrace alphabet and let $\equiv_{\{ser, inl\}}$, called *g-comtrace congruence*, be the EQ -congruence defined by the set of equations $EQ = EQ_1 \cup EQ_2$, where

$$EQ_1 \stackrel{df}{=} \{A = BC \mid A = B \cup C \in \mathbb{S} \wedge B \times C \subseteq ser\}, \\ EQ_2 \stackrel{df}{=} \{BA = AB \mid A \in \mathbb{S} \wedge B \in \mathbb{S} \wedge A \times B \subseteq inl\}.$$

The equational monoid $(\mathbb{S}^* / \equiv_{\{ser, inl\}}, \otimes, [\lambda])$ is called a *monoid of g-comtraces* over Θ . ■

Since ser and inl are irreflexive, $(A = BC) \in EQ_1$ implies $B \cap C = \emptyset$, and $(AB = BA) \in EQ_2$ implies $A \cap B = \emptyset$. Since $inl \cap sim = \emptyset$, we also have that if $(AB = BA) \in EQ_2$, then $A \cup B \notin \mathbb{S}$.

By Proposition 2.1, g-comtrace congruence relations can also be defined explicitly in non-equational form as follows.

Proposition 6.1. Let $\Theta = (E, \text{sim}, \text{ser}, \text{inl})$ be a g-comtrace alphabet and let \mathbb{S}^* be the set of all step sequences defined on Θ .

- Let $\approx_1 \subseteq \mathbb{S}^* \times \mathbb{S}^*$ be the relation comprising all pairs (t, u) of step sequences such that $t = wAz$ and $u = wBz$ where $w, z \in \mathbb{S}^*$ and A, B, C are steps satisfying $B \cup C = A$ and $B \times C \subseteq \text{ser}$.
- Let $\approx_2 \subseteq \mathbb{S}^* \times \mathbb{S}^*$ be the relation comprising all pairs (t, u) of step sequences such that $t = wABz$ and $u = wBAz$ where $w, z \in \mathbb{S}^*$ and A, B are steps satisfying $A \times B \subseteq \text{inl}$.

We define $\approx_{\{\text{ser}, \text{inl}\}} \stackrel{\text{df}}{=} \approx_1 \cup \approx_2$. Then $\equiv_{\{\text{ser}, \text{inl}\}} = \left(\approx_{\{\text{ser}, \text{inl}\}} \cup \approx_{\{\text{ser}, \text{inl}\}}^{-1} \right)^*$. \square

The name “generalized comtraces” comes from that fact that when $\text{inl} = \emptyset$, Definition 6.2 coincides with Definition 5.2 of comtrace monoids. Hence comtraces can be regarded as a special case of generalized comtraces. We will omit the subscript $\{\text{ser}, \text{inl}\}$ from the g-comtrace congruence and $\approx_{\{\text{ser}, \text{inl}\}}$, and write \equiv and \approx if it causes no ambiguity.

Example 6.2. The set \mathbf{x} from Example 6.1 is a g-comtrace with $E = \{a, b, c\}$, $\text{ser} = \text{sim} = \{(a, c), (c, a), (b, c), (c, b)\}$, $\text{inl} = \{(a, b), (b, a)\}$, and $\mathbb{S} = \{\{a, c\}, \{b, c\}, \{a\}, \{b\}, \{c\}\}$. So we write $\mathbf{x} = [\{a, c\}\{b\}]$. \blacksquare

It is worth noticing that there is an *important difference* between the equation $ab = ba$ for traces, and the equation $\{a\}\{b\} = \{b\}\{a\}$ for g-comtrace monoids. For traces, the equation $ab = ba$, when translated into step sequences, corresponds to two equations $\{a, b\} = \{a\}\{b\}$ and $\{a, b\} = \{b\}\{a\}$, which implies $\{a\}\{b\} \equiv \{a, b\} \equiv \{b\}\{a\}$. For g-comtrace monoids, the equation $\{a\}\{b\} = \{b\}\{a\}$ implies that $\{a, b\}$ is not a step, i.e., neither the equation $\{a, b\} = \{a\}\{b\}$ nor the equation $\{a, b\} = \{b\}\{a\}$ belongs to the set of equations. In other words, for traces the equation $ab = ba$ means ‘independency’, i.e., executing a and b in any order or simultaneously will yield the same consequence. For g-comtrace monoids, the equation $\{a\}\{b\} = \{b\}\{a\}$ means that execution of a and b in any order yields the same result, but executing of a and b in any order is *not* equivalent to executing them simultaneously.

7. Algebraic Properties of Comtrace and Generalized Comtrace Congruences

Algebraic properties of trace congruence operations such as *left/right cancellation* and *projection* are well understood [24]. They are intuitive and powerful tools with many applications [5]. In this section we will present equivalent properties for both comtrace and g-comtrace congruence. The basic obstacle is switching from sequences to step sequences.

7.1. Properties of Comtrace Congruence

Let us consider a comtrace alphabet $\theta = (E, \text{sim}, \text{ser})$ where we reserve \mathbb{S} to denote the set of all possible steps of θ throughout this section.

For each step sequence or enumerated step sequence $x = X_1 \dots X_k$, we define the *step sequence weight* of x as $\text{weight}(x) \stackrel{\text{df}}{=} \sum_{i=1}^k |X_i|$. We also define $\uplus(x) \stackrel{\text{df}}{=} \bigcup_{i=1}^k X_i$.

Due to the commutativity of the independency relation for traces, the *mirror rule*, which says if two sequences are congruent, then their *reverses* are also congruent, holds for *trace congruence* [5]. Hence, in trace theory, we only need a *right cancellation* operation to produce congruent *subsequences* from congruent sequences, since the *left cancellation* comes from the right cancellation of the reverses.

However, the *mirror rule* does not hold for comtrace congruence since the relation *ser* is usually not commutative. Example 5.1 works as a counter example since $\{a\}\{b,c\} \equiv \{a\}\{b\}\{c\}$ but $\{b,c\}\{a\} \not\equiv \{c\}\{b\}\{a\}$. Thus, we define separate left and right cancellation operators for comtraces.

Let $a \in E$, $A \in \mathbb{S}$ and $w \in \mathbb{S}^*$. The operator \div_R , *step sequence right cancellation*, is defined as follows:

$$\lambda \div_R a \stackrel{df}{=} \lambda, \quad wA \div_R a \stackrel{df}{=} \begin{cases} (w \div_R a)A & \text{if } a \notin A \\ w & \text{if } A = \{a\} \\ w(A \setminus \{a\}) & \text{otherwise.} \end{cases}$$

Symmetrically, a *step sequence left cancellation* operator \div_L is defined as follows:

$$\lambda \div_L a \stackrel{df}{=} \lambda, \quad Aw \div_L a \stackrel{df}{=} \begin{cases} A(w \div_L a) & \text{if } a \notin A \\ w & \text{if } A = \{a\} \\ (A \setminus \{a\})w & \text{otherwise.} \end{cases}$$

Finally, for each $D \subseteq E$, we define the function $\pi_D : \mathbb{S}^* \rightarrow \mathbb{S}^*$, *step sequence projection* onto D , as follows:

$$\pi_D(\lambda) \stackrel{df}{=} \lambda, \quad \pi_D(wA) \stackrel{df}{=} \begin{cases} \pi_D(w) & \text{if } A \cap D = \emptyset \\ \pi_D(w)(A \cap D) & \text{otherwise.} \end{cases}$$

The below result shows that the algebraic properties of comtraces are similar to the algebraic properties of traces [24].

Proposition 7.1.

1. $u \equiv v \implies \text{weight}(u) = \text{weight}(v)$. (step sequence weight equality)
2. $u \equiv v \implies |u|_a = |v|_a$. (event-preserving)
3. $u \equiv v \implies u \div_R a \equiv v \div_R a$. (right cancellation)
4. $u \equiv v \implies u \div_L a \equiv v \div_L a$. (left cancellation)
5. $u \equiv v \iff \forall s, t \in \mathbb{S}^*. sut \equiv svt$. (step subsequence cancellation)
6. $u \equiv v \implies \pi_D(u) \equiv \pi_D(v)$. (projection rule)

PROOF. For all except (5), it suffices to show that $u \approx v$ implies the right hand side of (1)–(6). Note that $u \approx v$ means $u = xAy$, $v = xBCy$, where $A = B \cup C$, $B \cap C = \emptyset$ and $B \times C \subseteq \text{ser}$.

1. Since $A = B \cup C$ and $B \cap C = \emptyset$, we have $\text{weight}(A) = |A| = |B| + |C| = \text{weight}(BC)$. Hence, $\text{weight}(u) = \text{weight}(x) + \text{weight}(A) + \text{weight}(z) = \text{weight}(x) + \text{weight}(BC) + \text{weight}(z) = \text{weight}(v)$.
2. There are two cases:
 - $a \in A$: Then $a \notin B \cap C$ because $B \cap C = \emptyset$. Since $A = B \cup C$, either $a \in B$ or $a \in C$. Then $|A|_a = |BC|_a$. Therefore, $|u|_a = |x|_a + |A|_a + |z|_a = |x|_a + |BC|_a + |z|_a = |v|_a$.

- $a \notin A$: Since $A = B \cup C$, $a \notin B \wedge a \notin C$. So $|A|_a = |BC|_a = 0$. Therefore, $|u|_a = |x|_a + |z|_a = |v|_a$.

3. There are four cases:

- $a \in \mathcal{J}(y)$: Let $z = y \div_R a$. Then $u \div_R a = xAz \approx xBCz = v \div_R a$.
- $a \notin \mathcal{J}(y)$, $a \in A \cap C$: Then $u \div_R a = x(A \setminus \{a\})y \approx xB(C \setminus \{a\})y = v \div_R a$.
- $a \notin \mathcal{J}(y)$, $a \in A \cap B$: Then $u \div_R a = x(A \setminus \{a\})y \approx x(B \setminus \{a\})Cy = v \div_R a$.
- $a \notin \mathcal{J}(Ay)$: Let $z = x \div_R a$. Then $u \div_R a = zAy \approx zBCy = v \div_R a$.

4. Dually to (3).

5. (\Rightarrow) Follows from the fact that \equiv is a congruence.

(\Leftarrow) For any two step sequences $s, t \in \mathbb{S}^*$, since $sut \equiv svt$, it follows that $(sut \div_R t) \div_L s = u \equiv v = (svt \div_R t) \div_L s$. Therefore, $u \equiv v$.

6. Note that $\pi_D(A) = \pi_D(B) \cup \pi_D(C)$ and $\pi_D(B) \times \pi_D(C) \subseteq \text{ser}$. So $\pi_D(u) = \pi_D(x)\pi_D(A)\pi_D(y) \approx \pi_D(x)\pi_D(B)\pi_D(C)\pi_D(y) = \pi_D(v)$. \square

Note that $(w \div_R a) \div_R b = (w \div_R b) \div_R a$, so we define

$$w \div_R \{a_1, \dots, a_k\} \stackrel{df}{=} \left(\dots ((w \div_R a_1) \div_R a_2) \dots \right) \div_R a_k, \text{ and}$$

$$w \div_R A_1 \dots A_k \stackrel{df}{=} \left(\dots ((w \div_R A_1) \div_R A_2) \dots \right) \div_R A_k$$

We define dually for \div_L . Hence Proposition 7.1 (4) and (5) can be generalized as follows.

Corollary 7.1. *For all $u, v, x \in \mathbb{S}^*$, we have*

1. $u \equiv v \implies u \div_R x \equiv v \div_R x$.
2. $u \equiv v \implies u \div_L x \equiv v \div_L x$. \square

7.2. Properties of Generalized Comtrace Congruence

We now show that g-comtrace congruence has virtually the same algebraic properties as comtrace congruence.

Proposition 7.2. *Let \mathbb{S} be the set of all steps over a g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$ and $u, v \in \mathbb{S}^*$. Then*

1. $u \equiv v \implies \text{weight}(u) = \text{weight}(v)$. (step sequence weight equality)
2. $u \equiv v \implies |u|_a = |v|_a$. (event-preserving)
3. $u \equiv v \implies u \div_R a \equiv v \div_R a$. (right cancellation)
4. $u \equiv v \implies u \div_L a \equiv v \div_L a$. (left cancellation)
5. $u \equiv v \iff \forall s, t \in \mathbb{S}^*. sut \equiv svt$. (step subsequence cancellation)
6. $u \equiv v \implies \pi_D(u) \equiv \pi_D(v)$. (projection rule)

PROOF. For all except (5), it suffices to show that $u \approx v$ implies the right hand side of (1)–(6). Notice that when $u \approx v$, the case $u = xAy \approx v = xBCy$ follows from Proposition 7.1. So we only need to consider the case $u = xAB y$ and $v = xBA y$, where $A \times B \subseteq \text{inl}$ and $A \cap B = \emptyset$. The rest can be proved similarly to the proof of Proposition 7.1. \square

Corollary 7.2. For all step sequences u, v, x over a g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$,

1. $u \equiv v \implies u \div_R x \equiv v \div_R x$.
2. $u \equiv v \implies u \div_L x \equiv v \div_L x$. □

The following proposition ensures that if any relation from the set $\{\leq, \geq, <, >, =, \neq\}$ between the positions of two event occurrences holds *after applying cancellation or projection operations* on a g-comtrace $[\bar{u}]$, then it also holds for the whole $[\bar{u}]$. In other words, both cancellation and projection preserve ordering in the stratified orders defined by g-comtraces.

Proposition 7.3. Let \bar{u} be an enumerated step sequence over a g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$ and $\alpha, \beta, \gamma \in \Sigma_u$ such that $\gamma \notin \{\alpha, \beta\}$. Let $\mathcal{R} \in \{\leq, \geq, <, >, =, \neq\}$. Then

1. If $\forall \bar{v} \in [\bar{u} \div_L \gamma]. \text{pos}_{\bar{v}}(\alpha) \mathcal{R} \text{pos}_{\bar{v}}(\beta)$, then $\forall \bar{w} \in [\bar{u}]. \text{pos}_{\bar{w}}(\alpha) \mathcal{R} \text{pos}_{\bar{w}}(\beta)$.
2. If $\forall \bar{v} \in [\bar{u} \div_R \gamma]. \text{pos}_{\bar{v}}(\alpha) \mathcal{R} \text{pos}_{\bar{v}}(\beta)$, then $\forall \bar{w} \in [\bar{u}]. \text{pos}_{\bar{w}}(\alpha) \mathcal{R} \text{pos}_{\bar{w}}(\beta)$.
3. If $S \subseteq \Sigma_u$ such that $\{\alpha, \beta\} \subseteq S$, then

$$(\forall \bar{v} \in [\pi_S(\bar{u})]. \text{pos}_{\bar{v}}(\alpha) \mathcal{R} \text{pos}_{\bar{v}}(\beta)) \implies (\forall \bar{w} \in [\bar{u}]. \text{pos}_{\bar{w}}(\alpha) \mathcal{R} \text{pos}_{\bar{w}}(\beta)).$$

PROOF. 1. Assume that

$$\forall \bar{v} \in [\bar{v} \div_L \gamma]. \text{pos}_{\bar{v}}(\alpha) \mathcal{R} \text{pos}_{\bar{v}}(\beta) \quad (7.1)$$

Suppose for a contradiction that $\exists \bar{w} \in [\bar{v}]. \neg(\text{pos}_{\bar{w}}(\alpha) \mathcal{R} \text{pos}_{\bar{w}}(\beta))$. Since $\gamma \notin \{\alpha, \beta\}$, we have $\neg(\text{pos}_{\bar{w} \div_L \gamma}(\alpha) \mathcal{R} \text{pos}_{\bar{w} \div_L \gamma}(\beta))$. But $\bar{w} \in [\bar{v}]$ implies $\bar{w} \div_L \gamma \equiv \bar{u} \div_L \gamma$. Hence, $\bar{w} \div_L \gamma \in [\bar{u} \div_L \gamma]$ and $\neg(\text{pos}_{\bar{w} \div_L \gamma}(\alpha) \mathcal{R} \text{pos}_{\bar{w} \div_L \gamma}(\beta))$, contradicting (7.1).

2. Dually to part (1).
3. Assume that

$$\forall \bar{v} \in [\pi_S(\bar{u})]. \text{pos}_{\bar{v}}(\alpha) \mathcal{R} \text{pos}_{\bar{v}}(\beta) \quad (7.2)$$

Suppose for a contradiction that $\exists \bar{w} \in [\bar{v}]. \neg(\text{pos}_{\bar{w}}(\alpha) \mathcal{R} \text{pos}_{\bar{w}}(\beta))$. Since $\{\alpha, \beta\} \subseteq S$, we have $\neg(\text{pos}_{\pi_S(\bar{w})}(\alpha) \mathcal{R} \text{pos}_{\pi_S(\bar{w})}(\beta))$. But $\bar{w} \in [\bar{v}]$ implies $\pi_S(\bar{w}) \equiv \pi_S(\bar{u})$. Hence, $\pi_S(\bar{w}) \in [\pi_S(\bar{u})]$ and $\neg(\text{pos}_{\pi_S(\bar{w})}(\alpha) \mathcal{R} \text{pos}_{\pi_S(\bar{w})}(\beta))$, contradicting (7.2). □

Clearly the above results also holds for comtraces as they are just g-comtraces with $\text{inl} = \emptyset$.

8. Maximally Concurrent and Canonical Representations

We will show that both traces, comtraces and g-comtraces have some special representations, that intuitively correspond to *maximally concurrent execution of concurrent histories*, i.e., “executing as much as possible in parallel”⁷. However such representations are truly unique only for comtraces. For traces and g-comtraces unique (or *canonical*) representations are obtained by adding some arbitrary total ordering on their alphabets.

In this section we will start with the general case of g-comtraces and then will consider comtraces and traces as special and more regular cases.

⁷This kind of semantics is formally defined and analyzed for example in [4].

8.1. Representations of Generalized Comtraces

Let $\Theta = (E, \text{sim}, \text{ser}, \text{inl})$ be a g-comtrace alphabet and \mathbb{S} be the set of all steps over Θ . We will start with the most “natural” definition which is the straightforward application of the approach used in [4] for an alternative version of traces called “vector firing sequences” (see [15, 27]).

Definition 8.1 (Greedy maximally concurrent form). A step sequence $u = A_1 \dots A_k \in \mathbb{S}^*$ is in *greedy maximally concurrent form (GMC-form)* if and only if for each $i = 1, \dots, k$:

$$(B_i y_i \equiv A_i \dots A_k) \implies |B_i| \leq |A_i|,$$

where for all $i = 1, \dots, k$, $A_i, B_i \in \mathbb{S}$, and $y_i \in \mathbb{S}^*$. ■

Proposition 8.1. For each g-comtrace \mathbf{u} over Θ there is a step sequence $u \in \mathbb{S}^*$ in GMC-form such that $\mathbf{u} = [u]$.

PROOF. Let $u = A_1 \dots A_k$, where the steps A_1, \dots, A_k are generated by the following simple greedy algorithm:

- 1: Initialize $i \leftarrow 0$ and $v_0 \leftarrow v$
- 2: **while** $v_i \neq \lambda$ **do**
- 3: $i \leftarrow i + 1$
- 4: Find A_i such that for each $B_i y_i \equiv v_{i-1}$, $|B_i| \leq |A_i|$
- 5: $v_i \leftarrow v_{i-1} \div_L A_i$
- 6: **end while**
- 7: $k \leftarrow i - 1$.

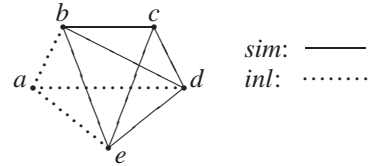
Since $\text{weight}(v_{i+1}) < \text{weight}(v_i)$ the above algorithm always terminates. Clearly $u = A_1 \dots A_k$ is in GMC-form and $u \in \mathbf{u}$. □

The algorithm from the proof of Proposition 8.1 used to generate A_1, \dots, A_k justifies the prefix “greedy” in Definition 8.1. However the GMC representation of g-comtraces is seldom unique and often intuitively “not maximally concurrent”. Consider the following two examples.

Example 8.1. Let $E = \{a, b, c\}$, $\text{sim} = \{(a, c), (c, a)\}$, $\text{ser} = \text{sim}$ and $\text{inl} = \{(a, b), (b, a)\}$ and $\mathbf{u} = [\{a\}\{b\}\{c\}] = \{\{a\}\{b\}\{c\}, \{b\}\{a\}\{c\}, \{b\}\{a, c\}\}$. Note that every element of \mathbf{u} is in GMC-form, but only $\{b\}\{a, c\}$ can intuitively be interpreted as maximally concurrent. ■

Example 8.2.

Let $E = \{a, b, c, d, e\}$, and $\text{sim} = \text{ser}$, inl be as in the picture on the right, and let $\mathbf{u} = [\{a\}\{b, c, d, e\}]$. One can easily verify by inspection that $\{a\}\{b, c, d, e\}$ is the shortest element of \mathbf{u} and the only element of \mathbf{u} in GMC-form is $\{b, e, d\}\{a\}\{c\}$. The step sequence $\{b, e, d\}\{a\}\{c\}$ is longer and intuitively less maximally concurrent than the step sequence $\{a\}\{b, c, d, e\}$. ■



Hence for g-comtraces the greedy maximal concurrency notion is not necessarily the global maximal concurrency notion, so we will try another approach.

Let $x = A_1 \dots A_k$ be a step sequence. We define $\text{length}(A_1 \dots A_k) \stackrel{df}{=} k$.

We also say that A_i is *maximally concurrent in x* if $B_i y_i \equiv A_i \dots A_k \implies |B_i| \leq |A_i|$.

Note that A_k is *always maximally concurrent in x* , which makes the following definition correct.

For every step sequence $x = A_1 \dots A_k$, let $mc(x)$ be the smallest i such that A_i is maximally concurrent in x .

Definition 8.2. A step sequence $u = A_1 \dots A_k$ is *maximally concurrent (MC-)* if and only if

1. $v \equiv u \implies \text{length}(u) \leq \text{length}(v)$,
2. for all $i = 1, \dots, k$ and for all w ,

$$(u_i = A_i \dots A_k \equiv w \wedge \text{length}(u_i) = \text{length}(w)) \implies mc(u_i) \leq mc(w).$$

■

Theorem 8.1. For every g-comtrace \mathbf{u} , there exists a step sequence $u \in \mathbf{u}$ such that u is maximally concurrent.

PROOF. Let $u_1 \in \mathbf{u}$ be a step sequence such that for each v , $v \equiv u_1 \implies \text{length}(u_1) \leq \text{length}(v)$, and $(v \equiv u_1 \wedge \text{length}(u_1) = \text{length}(v)) \implies mc(u_1) \leq mc(v)$. Obviously such u_1 exists for every g-comtrace \mathbf{u} . Assume that $u_1 = A_1 w_1$ and $\text{length}(u_1) = k$. Let u_2 be a step sequence satisfying $u_2 \equiv w_1$, $u_2 \equiv v \implies \text{length}(u_2) \leq \text{length}(v)$, and $(v \equiv u_2 \wedge \text{length}(u_2) = \text{length}(v)) \implies mc(u_2) \leq mc(v)$. Assume that $u_2 = A_2 w_3$. We repeat this process $k - 1$ times. Note that $u_k = A_k \in \mathbb{S}$. The step sequence $u = A_1 \dots A_k$ is maximally concurrent and $u \in \mathbf{u}$. □

For the case of Example 8.1 the step sequence $\{b\}\{a, c\}$ is maximally concurrent and for the case of Example 8.2 the step sequence $\{a\}\{b, c, d, e\}$ is maximally concurrent. There may be more than one maximally concurrent step sequences in a g-comtrace. For example if $E = \{a, b\}$, $\text{sim} = \text{ser} = \emptyset$, $\text{inl} = \{(a, b), (b, a)\}$, then the g-comtrace $t = [\{a\}\{b\}] = \{\{a\}\{b\}, \{b\}\{a\}\}$ and both $\{a\}\{b\}$ and $\{b\}\{a\}$ are maximally concurrent.

Having a unique representation is often very useful in proving properties about g-comtraces since it allows us to *uniquely identify* a g-comtrace. Furthermore, to be really useful in proofs, a unique representation must also have an easy to handle definition. For g-comtraces we can get unique representation by introducing some total ordering of steps and then apply this ordering on either GMC-forms or MC-forms. To achieve this purpose, we just need an arbitrary total order on the set of events E . However the definition of GMC-form is local (step by step) and easier to handle than the definition of MC-form, which is global as it requires comparing the lengths of all step sequences in a given g-comtrace. Because of “greediness”, ordering different GMC-representations is also simpler than ordering different MC-representations. Hence, we will base our unique representation on the idea of GMC-form.

Definition 8.3 (Lexicographical ordering). Assume that we have a *total order* $<_E$ on E .

1. We define a *step order* $<^{st}$ on \mathbb{S} as follows:

$$A <^{st} B \stackrel{df}{\iff} |A| > |B| \vee (|A| = |B| \wedge A \neq B \wedge \min_{<_E}(A \setminus B) <_E \min_{<_E}(B \setminus A)),$$

where $\min_{<_E}(X)$ denotes the least element of the set $X \subseteq E$ w.r.t. $<_E$.

2. Let $A_1 \dots A_n$ and $B_1 \dots B_m$ be two sequences in \mathbb{S}^* . We define a *lexicographical order* $<^{lex}$ on step sequences in a natural way as the lexicographical order induced by $<^{st}$, i.e.,

$$A_1 \dots A_n <^{lex} B_1 \dots B_m \stackrel{df}{\iff} \exists k > 0. \left((\forall i < k. A_i = B_i) \wedge (A_k <^{st} B_k \vee n < k \leq m) \right).$$

■

Directly from the above definition we have the desired properties of $<^{st}$ and $<^{lex}$.

Corollary 8.1.

1. The step order $<^{st}$ is a total order on \mathbb{S} .
2. The lexicographical order $<^{lex}$ is a total order on \mathbb{S}^* .

□

Example 8.3. Assume that $a <_E b <_E c <_E d <_E e$. Then we have $\{a, b, c, e\} <^{st} \{b, c, d\}$ since $\{a, b, c, e\} \setminus \{b, c, d\} = \{a\}$, $\{b, c, d\} \setminus \{a, b, c, e\} = \{d\}$, and $a <_E d$. And $\{a, c\}\{b, c\}\{d\}\{d, c\} <^{lex} \{a, c\}\{b\}\{c, d, e\}$ since $|\{b, c\}| > |\{b\}|$. ■

Definition 8.4 (g-Canonical step sequence). A step sequence $x \in \mathbb{S}^*$ is *g-canonical* if for every step sequence $y \in \mathbb{S}^*$, we have $(x \equiv y \wedge x \neq y) \implies x <^{lex} y$. ■

In other words, x is g-canonical if it is the least element in the g-comtrace $[x]$ with respect to the lexicographical ordering $<^{lex}$.

Corollary 8.2.

1. Each g-canonical step-sequence is in GMC-form.
2. For every step sequence $x \in \mathbb{S}^*$, there exists exactly one g-canonical sequence u such that $x \equiv u$.

□

All of the concepts and results discussed so far in this section hold also for general equational monoids derived from the step sequence monoid (like those considered in [16]).

We will now show that for both comtraces and traces, the GMC-form, MC-form and g-canonical form correspond to the canonical form discussed in [2, 4, 12, 16].

8.2. Canonical Representations of Comtraces

First note that comtraces are just g-comtraces with *inl* being empty relation, so all definitions for g-comtraces also hold for comtraces.

Let $\theta = (E, \text{sim}, \text{ser})$ be a comtrace alphabet (i.e. *inl* = \emptyset) and \mathbb{S} be the set of all steps over θ . In principle, $(a, b) \in \text{ser}$ means that the sequence $\{a\}\{b\}$ can be replaced by the set $\{a, b\}$ (and vice versa). We start with the definition of a relation between steps that allow such replacement.

Definition 8.5 (Forward dependency). Let $\mathbb{FD} \subseteq \mathbb{S} \times \mathbb{S}$ be a relation comprising all pairs of steps (A, B) such there exists a step $C \in \mathbb{S}$ such that

$$C \subseteq B \wedge A \times C \subseteq \text{ser} \wedge C \times (B \setminus C) \subseteq \text{ser}.$$

The relation \mathbb{FD} is called *forward dependency* on steps. ■

Note that in this definition $C \in \mathbb{S}$ implies $C \neq \emptyset$, but $C = B$ is allowed. The next result explains the name “forward dependency” of \mathbb{FD} . If $(A, B) \in \mathbb{FD}$, then some elements from B can be moved to A and the outcome will still be equivalent to AB .

Lemma 8.1. $(A, B) \in \mathbb{FD} \iff (\exists C \in \widehat{\mathcal{P}}(B). (A \cup C)(B \setminus C) \equiv AB) \vee A \cup B \equiv AB$.

PROOF. (\Rightarrow) If $C = B$ then $A \cup B \approx AB$ which implies $A \cup B \equiv AB$. If $C \subset B$ and $C \neq \emptyset$ then we have $(A \cup C)(B \setminus C) \approx AC(B \setminus C) \approx AB$, i.e. $(A \cup C)(B \setminus C) \equiv AB$.

(\Leftarrow) Assume $A \cup B \equiv AB$. This means $A \cup B \in \mathbb{S}$ and consequently $A \cap B = \emptyset$, $A \times C \subseteq \text{ser}$. Let $a \in A$, $b \in B$. By Proposition 7.1(6), $\{a, b\} = \pi_{\{a, b\}}(A \cup B) \equiv \pi_{\{a, b\}}(AB) = \{a\}\{b\}$. But $\{a, b\} \equiv \{a\}\{b\}$ means $(a, b) \in \text{ser}$. Therefore $A \times B \subseteq \text{ser}$, i.e. $(A, B) \in \mathbb{FD}$.

Assume $C \subset B$, $C \neq \emptyset$ and $(A \cup C)(B \setminus C) \equiv AB$. This implies $A \cup C \in \mathbb{S}$ and $A \cap C = \emptyset$. Let $a \in A$ and $c \in C$. By Proposition 7.1(6), $\{a, c\} = \pi_{\{a, c\}}(A \cup C)(B \setminus C) \equiv \pi_{\{a, c\}}(AB) = \{a\}\{c\}$. But $\{a, c\} \equiv \{a\}\{c\}$ means $(a, c) \in \text{ser}$. Hence $A \times C \subseteq \text{ser}$. Let $b \in B \setminus C$ and $c \in C$. By Proposition 7.1(6), $\{c\}\{b\} = \pi_{\{b, c\}}(A \cup C)(B \setminus C) \equiv \pi_{\{b, c\}}(AB) = \{b, c\}$. Thus $\{c\}\{b\} \equiv \{b, c\}$, which means $(c, b) \in \text{ser}$, i.e. $C \times (B \setminus C) \subseteq \text{ser}$. Hence $(A, B) \in \mathbb{FD}$. \square

We will now recall the definition of a canonical step sequence for comtraces.

Definition 8.6 (Comtrace canonical step sequence [12]). A step sequence $u = A_1 \dots A_k$ is *canonical* if we have $(A_i, A_{i+1}) \notin \mathbb{FD}$ for all i , $1 \leq i < k$. \blacksquare

The next results shows that the canonical step sequence for comtraces is in fact “greedy”.

Lemma 8.2. For each canonical step sequence $u = A_1 \dots A_k$, we have

$$A_1 = \{a \mid \exists w \in [u]. w = C_1 \dots C_m \wedge a \in C_1\}.$$

PROOF. Let $A = \{a \mid \exists w \in [u]. w = C_1 \dots C_m \wedge a \in C_1\}$. Since $u \in [u]$, $A_1 \subseteq A$. We need to prove that $A \subseteq A_1$. Definitely $A = A_1$ if $k = 1$, so assume $k > 1$. Suppose that $a \in A \setminus A_1$, $a \in A_j$, $1 < j \leq k$ and $a \notin A_i$ for $i < j$. Since $a \in A$, there is $v = Bx \in [u]$ such that $a \in B$. Note that $A_{j-1}A_j$ is also *canonical* and $u' = A_{j-1}A_j = (u \div_R (A_{j+1} \dots A_k)) \div_L (A_1 \dots A_{j-2})$. Let $v' = (v \div_R (A_{j+1} \dots A_k)) \div_L (A_1 \dots A_{j-2})$. We have $v' = B'x'$ where $a \in B'$. By Corollary 7.1, $u' \equiv v'$. Since $u' = A_{j-1}A_j$ is canonical then $\exists c \in A_{j-1}$. $(c, a) \notin \text{ser}$ or $\exists b \in A_j$. $(a, b) \notin \text{ser}$.

- For the former case: $\pi_{\{a, c\}}(u') = \{c\}\{a\}$ (if $c \notin A_j$) or $\pi_{\{a, c\}}(u') = \{c\}\{a, c\}$ (if $c \in A_j$). If $\pi_{\{a, c\}}(u') = \{c\}\{a\}$ then $\pi_{\{a, c\}}(v')$ equals either $\{a, c\}$ (if $c \in B'$) or $\{a\}\{c\}$ (if $c \notin B'$), i.e., in both cases $\pi_{\{a, c\}}(u') \neq \pi_{\{a, c\}}(v')$, contradicting Proposition 7.1(6). If $\pi_{\{a, c\}}(u') = \{c\}\{a, c\}$ then $\pi_{\{a, c\}}(v')$ equals either $\{a, c\}\{c\}$ (if $c \in B'$) or $\{a\}\{c\}\{c\}$ (if $c \notin B'$). However in both cases $\pi_{\{a, c\}}(u') \neq \pi_{\{a, c\}}(v')$, contradicting Proposition 7.1(6). For the latter case, let $d \in A_{j-1}$. Then $\pi_{\{a, b, d\}}(u') = \{d\}\{a, b\}$ (if $d \notin A_j$), or $\pi_{\{a, b, d\}}(u') = \{d\}\{a, b, d\}$ (if $d \in A_j$). If $\pi_{\{a, b, d\}}(u') = \{d\}\{a, b\}$ then $\pi_{\{a, b, d\}}(v')$ is one of the following $\{a, b, d\}$, $\{a, b\}\{d\}$, $\{a, d\}\{b\}$, $\{a\}\{b\}\{d\}$ or $\{a\}\{d\}\{b\}$, and in either case $\pi_{\{a, b, d\}}(u') \neq \pi_{\{a, b, d\}}(v')$, again contradicting Proposition 7.1(6).
- If $\pi_{\{a, b, d\}}(u') = \{d\}\{a, b, d\}$, then we know $\pi_{\{a, b, d\}}(v')$ is one of the following $\{a, b, d\}\{d\}$, $\{a, b\}\{d\}\{d\}$, $\{a, d\}\{b, d\}$, $\{a, d\}\{b\}\{d\}$, $\{a, d\}\{d\}\{b\}$, $\{a\}\{b\}\{d\}\{d\}$, $\{a\}\{d\}\{b\}\{d\}$, or $\{a\}\{d\}\{d\}\{b\}$. However in any of these cases we have $\pi_{\{a, b, d\}}(u') \neq \pi_{\{a, b, d\}}(v')$, contradicting Proposition 7.1(6) as well.

□

We will now show that for comtraces the canonical form as defined by Definition 8.6 and GMC-form are equivalent, and that each comtrace has a unique canonical representation.

Theorem 8.2. *A step sequence u is in GMC-form if and only if it is canonical.*

PROOF. (\Leftarrow) Suppose that $u = A_1 \dots A_k$ is canonical. By Lemma 8.2 we have that for each $B_1 y_1 \equiv A_1 \dots A_k$, $|B_1| \leq |A_1|$. Since each $A_i \dots A_k$ is also canonical, $A_2 \dots A_k$ is canonical so by Lemma 8.2 again we have that for each $B_2 y_2 \equiv A_2 \dots A_k$, $|B_2| \leq |A_2|$. And so on, i.e. $u = A_1 \dots A_k$ is in GMC-form.

(\Rightarrow) Suppose that $u = A_1 \dots A_k$ is not canonical, and j is the smallest number such that $(A_j, A_{j+1}) \in \mathbb{FD}$. Hence $A_1 \dots A_{j-1}$ is canonical, and, by (1) of this Theorem, in GMC-form. By Lemma 8.1, either there is a non empty $C \subset A_{j+1}$ such that $(A_j \cup C)(A_{j+1} \setminus B) \equiv A_j A_{j+1}$, or $A_j \cup A_{j+1} \equiv A_j A_{j+1}$. In the first case since $C \neq \emptyset$, then $|A_j \cup C| > |A_j|$, in the second case $|A_j \cup A_{j+1}| > |A_j|$, so $A_j \dots A_k$ is not in GMC-form, which means $u = A_1 \dots A_k$ is not in GMC-form either. □

Theorem 8.3 (implicit in [12]). *For each step sequence v there is a unique canonical step sequence u such that $v \equiv u$.*

PROOF. The existence follows from Proposition 8.1 and Theorem 8.2. We only need to show uniqueness. Suppose that $u = A_1 \dots A_k$ and $v = B_1 \dots B_m$ are both canonical step sequences and $u \equiv v$. By induction on $k = |u|$ we will show that $u = v$. By Lemma 8.2, we have $B_1 = A_1$. If $k = 1$, this ends the proof. Otherwise, let $u' = A_2 \dots A_k$ and $w' = B_2 \dots B_m$ and u', v' are both canonical step sequences of $[u']$. Since $|u'| < |u|$, by the induction hypothesis, we obtain $A_i = B_i$ for $i = 2, \dots, k$ and $k = m$. □

The result of Theorem 8.3 was not stated explicitly in [12], but it can be derived from the results of Propositions 3.1, 4.8 and 4.9 of [12]. However Propositions 3.1 and 4.8 of [12] involve the concepts of partial orders and stratified order structures, while the proof of Theorem 8.3 uses only the algebraic properties of step sequences and comtraces.

Immediately from Theorems 8.2 and 8.3 we get the following result.

Corollary 8.3. *A step sequence u is canonical if and only if it is g-canonical.* □

It turns out that for comtraces the canonical representation and MC-representation are also equivalent.

Lemma 8.3. *If a step sequence u is canonical and $u \equiv v$, then $\text{length}(u) \leq \text{length}(v)$.*

PROOF. By induction on $\text{length}(v)$. Obvious for $\text{length}(v) = 1$ as then $u = v$. Assume it is true for all v such that $\text{length}(v) \leq r - 1$, $r \geq 2$. Consider $v = B_1 B_2 \dots B_r$ and let $u = A_1 A_2 \dots A_k$ be a canonical step sequence such that $v \equiv u$. Let $v_1 = v \div_L A_1 = C_1 \dots C_s$. By Corollary 7.1(2), $v_1 \equiv u \div_L A_1 = A_2 \dots A_k$, and $A_2 \dots A_k$ is clearly canonical. Hence by induction assumption $k - 1 = \text{length}(A_2 \dots A_k) \leq s$. By Lemma 8.2, $B_1 \subseteq A_1$, hence $v_1 = v \div_L A_1 = B_2 \dots B_r \div_L A_1 = C_1 \dots C_s$, which means $s \leq r - 1$. Therefore $k - 1 \leq s \leq r - 1$, i.e. $k \leq r$, which ends the proof. □

Theorem 8.4. *A step sequence u is maximally concurrent if and only if it is canonical.*

PROOF. (\Leftarrow) Let u be canonical. From Lemma 8.3 it follows the condition (1) of Definition 8.2 is satisfied. By Theorem 8.2, u is in GMC-form, so the condition (2) of Definition 8.2 is satisfied as well.

(\Rightarrow) By induction on $\text{length}(u)$. It is obviously true for $u = A_1$. Suppose it is true for $\text{length}(u) = k$. Let $u = A_1 A_2 \dots A_k A_{k+1}$ be maximally concurrent. The step sequence $A_2 \dots A_{k+1}$ is also maximally concurrent and canonical by the induction assumption. If $A_1 A_2 \dots A_{k+1}$ is not canonical, then $(A_1, A_2) \in \mathbb{FD}$. By Lemma 8.1, either there is non-empty $C \subset B$ such that $(A_1 \cup C)(A_2 \setminus C) \equiv A_1 A_2$, or $A_1 \cup A_2 \equiv A_1 B_2$. Hence either $(A_1 \cup C)(A_2 \setminus C) A_3 \dots A_{k+1} \equiv A_1 \dots A_{k+1} = u$ or $(A_1 \cup A_2) A_3 \dots A_{k+1} \equiv A_1 \dots A_{k+1} = u$. The former contradicts the condition (2) of Definition 8.2, the latter one contradicts the condition (1) of Definition 8.2, so u is not maximally concurrent, which means $(A_1, A_2) \notin \mathbb{FD}$, so $u = A_1 \dots A_{k+1}$ is canonical. \square

Summing up, as far as canonical representation is concerned, comtraces are very regular. All three forms for g-comtraces, GMC-form, MC-form and g-canonical form, collapse to one comtrace canonical form if $\text{inl} = \emptyset$.

8.3. Canonical Representations of Traces

We will show that the canonical representations of traces are conceptually the same as the canonical representations of comtraces. The differences are merely “syntactical”, as traces are sets of sequences, so “maximal concurrency” cannot be expressed explicitly, while comtraces are sets of step sequences.

Let (E, ind) be a trace alphabet and $(E^* / \equiv, \otimes, [\lambda])$ be a monoid of traces. A sequence $x = a_1 \dots a_k \in E^*$ is called *fully commutative* if $(a_i, a_j) \in \text{ind}$ for all $i \neq j$ and $i, j \in \{1, \dots, k\}$.

Corollary 8.4. *If $x = a_1 \dots a_k \in E^*$ is fully commutative and $y = a_{i_1} \dots a_{i_k}$ is any permutation of $a_1 \dots a_k$, then $x \equiv y$.* \square

The above corollary could be interpreted as saying that if $x = a_1 \dots a_k \in E^*$ is fully commutative then the set of events $\{a_1, \dots, a_k\}$ can be executed simultaneously.

A fully commutative sequence y is *maximal in $x \in E^*$* if either $x = ybz$, or $x = way$ or $x = waybz$, for some $a, b \in E$, $w, z \in E^*$ and neither yb nor az are fully commutative.

Lemma 8.4. *Each sequence x has a unique decomposition $x = x_1 \dots x_k$ such that each fully commutative x_i is maximal in x .*

PROOF. By induction on x . Obvious for $x = a \in E$. Assume it is true for x . Consider xa . We have $xa = x_1 \dots x_k a$ where $x_1 \dots x_k$ is a unique decomposition of x . If $x_n a$ is fully commutative then $x_1 \dots x_{k-1} x'_k$, where $x'_k = x_k a$ is the unique decomposition of xa . If $x_k a$ is not fully commutative, then $x = x_1 \dots x_k x_{k+1}$, where $x_{k+1} = a$ is the unique decomposition of xa . \square

Definition 8.7 (Greedy maximally concurrent form for traces [2, 4]). A sequence $x \in E^*$ is in *greedy maximally concurrent form (GMC-form)* if $x = \lambda$ or $x = x_1 \dots x_n$ such that

1. each x_i is fully commutative, for $i = 1, \dots, n$,
2. for each $1 \leq i \leq n-1$ and for each element a of x_{i+1} there exists an element b of x_i such that $a \neq b$ and $(a, b) \notin \text{ind}$. \blacksquare

Corollary 8.4 and Lemma 8.4 explain and justify the name. Often the form from the above definition is called “canonical” [4, 15, 16].

Theorem 8.5 ([2, 4]). *For every trace $t \in E^* / \equiv$, there exists $x \in E^*$ such that $t = [x]$ and x is in the GMC-form.* \square

The GMC-form as defined above is not unique, a trace may have more than one GMC representation. For instance the trace $t_1 = [abcbca]$ from Example 3.1 has four GMC representations: $abcbca$, $acbbca$, $abccba$, and $acbcba$. The GMC-form is however unique when traces are represented as *vector firing sequences*⁸ [4, 15, 27], where each fully commutative sequence is represented by a single unique vector of events (so the name “canonical” used in [4, 15] is justified). To get uniqueness in standard Mazurkiewicz trace formalism, it suffices to order fully commutative sequences. For example we may introduce an arbitrary total order on E , extend it lexicographically to E^* and add the condition that in the representation $x = x_1 \dots x_n$, each x_i is minimal w.r.t. the lexicographic ordering. The GMC-form with this additional condition is called *Foata canonical form*.

Theorem 8.6 ([2]). *Every trace has a unique representation in the Foata canonical form.* \square

We will now show the relationship between GMC-form for traces and GMC-form (or canonical form) for comtraces.

Define \mathbb{S} , the set of steps generated by (E, ind) as the set of all cliques of the graph the relation ind , and for each fully commutative sequence $x = a_1 \dots a_n$, let $st(x) = \{a_1, \dots, a_n\} \in \mathbb{S}$ be the step generated by x .

For each sequence x such that its maximal fully commutative composition is $x = x_1 \dots x_k$, define $x^{\{max\}} = st(x_1) \dots st(x_k) \in \mathbb{S}^*$, its *maximally concurrent step sequence representation*. The name is formally justified by the following result (which also follows implicitly from [4]).

Proposition 8.2.

1. *A sequence x is in GMC-form in (E, ind) if and only if the step sequence $x^{\{max\}}$ is in GMC-form (or canonical form) in (E, sim, ser) where $sim = ser = ind$.*
2. $[x]_{\equiv ind} \stackrel{t \rightsquigarrow c}{\equiv} [x^{\{max\}}]_{\equiv ser}$.

PROOF. 1. Let $x = x_1 \dots x_k$ be maximally fully commutative representation of x . If x is not in GMC-form then by (2) of Definition 8.7, there are x_i, x_{i+1} and $a, b \in E$ such that $a \in st(x_i)$, $b \in st(x_{i+1})$ and $(a, b) \in ind$. Since $ser = ind$ this means that $(st(x_i), st(x_{i+1})) \in \mathbb{FD}$, so $x^{\{max\}}$ is not canonical. Suppose that $x^{\{max\}}$ is not canonical, i.e. $(st(x_i), st(x_{i+1})) \in \mathbb{FD}$ for some i . This means there is a non-empty $C \subseteq st(x_{i+1})$ such that $st(x_i) \times C \subseteq ser$ and $C \times (st(x_{i+1}) \setminus C) \subseteq ser$. Let $a \in st(x_i)$ and $b \in C \subseteq st(x_{i+1})$. Since $ind = ser$, then $(a, b) \in ind$, so x is not in GMC-form.

⁸The vector firing sequences were introduced by Mike Shields in 1979 [27] as an alternative equivalent representation of Mazurkiewicz traces.

2. By the definition $[x]/\equiv_{ind} \stackrel{t \rightsquigarrow c}{=} [x^{\{\}}]_{\equiv_{ser}}$. Let $a_1 \dots a_n$ be a fully commutative sequence. Since $ser = ind$, $\{a_1\} \dots \{a_n\} \equiv_{ser} \{a_1, \dots, a_n\}$. Hence, for each sequence x , $x^{\{\}} \equiv_{ser} x^{\{max\}}$, i.e. $[x^{\{\}}]_{\equiv_{ser}} = [x^{\{max\}}]_{\equiv_{ser}}$. \square

Hence we have proved that the GMC-form (or canonical form) for comtraces and GMC-form for traces are semantically identical concepts. They both describe the greedy maximally concurrent semantics, which for both comtraces and traces is also the global maximally concurrent semantics.

9. Generalized Comtrace and its Languages

As for traces, we can easily extend the concepts of comtraces and g-comtraces to the level of languages, with similar potential applications. We will give only the definitions and the results for g-comtraces, as they are practically identical in both cases.

Let $\Theta = (E, sim, ser, inl)$ be a g-comtrace alphabet and \mathbb{S} be the set of all possible steps over Θ . Any subset L of \mathbb{S}^* is a *step sequence language* over Θ , while any subset \mathcal{L} of \mathbb{S}^*/\equiv is a *g-comtrace language* over Θ .

For any step sequence language L , we define a g-comtrace language $[L]_{\Theta}$ (or just $[L]$) as $[L] \stackrel{df}{=} \{[u] \mid u \in L\}$, and $[L]$ is called the g-comtrace language *generated* by L .

For any g-comtrace language \mathcal{L} , we define $\bigcup \mathcal{L} \stackrel{df}{=} \{u \mid [u] \in \mathcal{L}\}$. Given step sequence languages L_1, L_2 and g-comtrace languages $\mathcal{L}_1, \mathcal{L}_2$ over the alphabet Θ , the *composition of languages* are defined as following:

$$L_1 L_2 \stackrel{df}{=} \{s_1 * s_2 \mid s_1 \in L_1 \wedge s_2 \in L_2\} \quad \mathcal{L}_1 \mathcal{L}_2 \stackrel{df}{=} \{\mathbf{t}_1 \circledast \mathbf{t}_2 \mid \mathbf{t}_1 \in \mathcal{L}_1 \wedge \mathbf{t}_2 \in \mathcal{L}_2\}$$

(Recall $*$ and \circledast denote the operators for step sequence and g-comtrace monoids respectively.)

We let L^* and \mathcal{L}^* denote the *Kleene closure* of the step sequence language L and the g-comtrace language \mathcal{L} . We define $L^* \stackrel{df}{=} \bigcup_{n \geq 0} L^n$, where $L^0 \stackrel{df}{=} \{\lambda\}$ and $L^{n+1} \stackrel{df}{=} L^n L$. We define $\mathcal{L}^* \stackrel{df}{=} \bigcup_{n \geq 0} \mathcal{L}^n$, where $\mathcal{L}^0 \stackrel{df}{=} \{[\lambda]\}$ and $\mathcal{L}^{n+1} \stackrel{df}{=} \mathcal{L}^n \mathcal{L}$.

Since g-comtrace languages are sets, one can use the standard set operations: union, intersection, difference, etc. The following result is a direct consequence of the g-comtrace language definition and the properties of g-comtrace composition.

Proposition 9.1. *Let L, L_1, L_2 and L_i for $i \in I$ be step sequence languages, and let \mathcal{L} be a g-comtrace language. Then :*

1. $[\emptyset] = \emptyset$
2. $[L_1][L_2] = [L_1 L_2]$
3. $L_1 \subseteq L_2 \Rightarrow [L_1] \subseteq [L_2]$
4. $L \subseteq \bigcup [L]$
5. $\mathcal{L} = [\bigcup \mathcal{L}]$
6. $[L_1] \cup [L_2] = [L_1 \cup L_2]$
7. $\bigcup_{i \in I} [L_i] = [\bigcup_{i \in I} L_i]$
8. $[L]^* = [L^*]$.

PROOF. The proof is the same to the case of traces in [24]. \square

When $inl = \emptyset$, we have the case of comtrace languages. The languages of comtraces and g-comtraces provide a bridge between operational and structural semantics. In other words, if a

step sequence language L describes an operational semantics of a given concurrent system, we only need to derive $(E, \text{sim}, \text{ser}, \text{inl})$ from the system, and the gcomtrace (comtrace) language $[L]$ defines the structural semantics of the system.

Example 9.1. Consider the following simple concurrent system Priority, which comprises two sequential subsystems such that

- the first subsystem can cyclically engage in event a followed by event b ,
- the second subsystem can cyclically engage in event b or in event c ,
- the two systems synchronize by means of handshake communication,
- there is a priority constraint stating that if it is possible to execute event b then c must not be executed.

This example has often been analyzed in the literature (cf. [14]), usually under the interpretation that $a = \text{'Error Message'}$, $b = \text{'Stop And Restart'}$, and $c = \text{'Some Action'}$. It can be formally specified in various notations including *Priority* and *Inhibitor Nets* (cf. [10, 13]). Its operational semantics (easily found in any model) can be defined by the following language of step sequences

$$L_{\text{Priority}} \stackrel{\text{df}}{=} \text{Pref}((\{c\}^* \cup \{a\}\{b\} \cup \{a, c\}\{b\})^*),$$

where $\text{Pref}(L) \stackrel{\text{df}}{=} \bigcup_{w \in L} \{u \in L \mid \exists v. uv = w\}$ denotes the *prefix closure* of L .

The rules for deriving the comtrace alphabet $(E, \text{sim}, \text{ser})$ depend on the model, and for Priority, the set of possible steps is $\mathbb{S} = \{\{a\}, \{b\}, \{c\}, \{a, c\}\}$, and $\text{ser} = \{(c, a)\}$ and $\text{ser} = \{(a, c), (c, a)\}$. Then, $[L_{\text{Priority}}]$ defines the structural comtrace semantics of Priority. For instance, $[\{a, c\}\{b\}] = \{\{c\}\{a\}\{b\}, \{a, c\}\{b\}\} \in [L_{\text{Priority}}]$. ■

Remark 9.1. As opposed to the case of trace languages, we know very little about the properties of comtrace languages, not to mention the properties of g-comtrace languages. In particular deep results analogous to Zielonka's theorem [30] are unknown.

10. Comtraces and Stratified Order Structures

This section consists of two parts. In the first part we will recall the major result of [12] that shows how comtraces define appropriate so-structures. The second part contains the new result showing how arbitrary finite so-structures can be represented by comtraces. This problem was not analyzed in [12].

We will start with the definition of \diamond -closure construction that plays a substantial role in most of the applications of so-structures for modelling concurrent systems (cf. [12, 20]).

Definition 10.1 (Diamond closure of relational structures [12]).

Given a relational structure $S = (X, R_1, R_2)$, we define S^\diamond , the \diamond -closure of S , as

$$S^\diamond \stackrel{\text{df}}{=} (X, \prec_{R_1, R_2}, \sqsubset_{R_1, R_2}),$$

where $\prec_{R_1, R_2} \stackrel{\text{df}}{=} (R_1 \cup R_2)^* \circ R_1 \circ (R_1 \cup R_2)^*$ and $\sqsubset_{R_1, R_2} \stackrel{\text{df}}{=} (R_1 \cup R_2)^* \setminus \text{id}_X$. ■

The motivation behind the above definition is the following. For ‘reasonable’ R_1 and R_2 , the relational structure $(X, R_1, R_2)^\diamond$ should satisfy the axioms S1–S4 of the so-structure definition. Intuitively, \diamond -closure is a generalization of the transitive closure constructions for relations to so-structures. Note that if $R_1 = R_2$ then $(X, R_1, R_2)^\diamond = (X, R_1^+, R_1^+)$. The following result shows that the properties of \diamond -closure are very close to the appropriate properties of transitive closure.

Theorem 10.1 (Closure properties of \diamond -closure [12]).

Let $S = (X, R_1, R_2)$ be a relational structure.

1. If R_2 is irreflexive, then $S \subseteq S^\diamond$.
2. $(S^\diamond)^\diamond = S^\diamond$.
3. S^\diamond is a so-structure if and only if $\prec_{R_1, R_2} = (R_1 \cup R_2)^* \circ R_1 \circ (R_1 \cup R_2)^*$ is irreflexive.
4. If S is a so-structure, then $S = S^\diamond$. □

Each comtrace is a set of equivalent step sequences and each step sequence represents a stratified order, so a comtrace can be interpreted as a set of equivalent stratified orders. From the theory presented in Section 4 and the fact that comtrace satisfies paradigm π_3 , it follows that this set of orders should define a so-structure, which should be called a so-structure defined by a given comtrace. On the other hand, with respect to a comtrace alphabet, every comtrace can be uniquely generated from any step sequence it contains. Thus, we will show that given a step sequence u over a comtrace alphabet, *without analyzing any other elements of the comtrace $[u]$ except u itself*, we will be able to construct the same so-structure as the one defined by the whole comtrace. Formulations and proofs of such results are done in [12] and depend heavily on the \diamond -closure construction and its properties.

Let $\theta = (E, \text{sim}, \text{ser})$ be a comtrace alphabet, and let $u \in \mathbb{S}^*$ be a step sequence and let $\triangleleft_u \subseteq \Sigma_u \times \Sigma_u$ be the stratified order generated by u . Note that if $u \equiv w$ then $\Sigma_u = \Sigma_w$. Thus, for every comtrace $\mathbf{x} = [x] \in \mathbb{S}^* / \equiv$, we can define $\Sigma_{\mathbf{x}} = \Sigma_x$.

We will now show how the \diamond -closure operator is used to define a so-structure induced by a single step sequence u .

Definition 10.2. Let $u \in \mathbb{S}^*$. We define the relations $\prec_u, \sqsubset_u \subseteq \Sigma_u \times \Sigma_u$ as:

1. $\alpha \prec_u \beta \stackrel{df}{\iff} \alpha \triangleleft_u \beta \wedge (l(\alpha), l(\beta)) \notin \text{ser},$
2. $\alpha \sqsubset_u \beta \stackrel{df}{\iff} \alpha \triangleleft_u \beta \wedge (l(\beta), l(\alpha)) \notin \text{ser}.$ ■

Lemma 10.1 ([12, Lemma 4.7]). For all $u, v \in \mathbb{S}^*$, if $u \equiv v$, then $\prec_u = \prec_v$ and $\sqsubset_u = \sqsubset_v$. □

Definition 10.2 together with Lemma 10.1 describes two basic *local* invariants of the elements of Σ_u . The relation \prec_u captures the situation when α *always precedes* β , and the relation \sqsubset_u captures the situation when α *never follows* β .

Definition 10.3. Given a step sequence $u \in \mathbb{S}^*$ and its respective comtrace $\mathbf{u} = [u] \in \mathbb{S}^* / \equiv$. We define the relational structures $S^{\{u\}}$ and $S_{\mathbf{u}}$ as follows:

$$S^{\{u\}} \stackrel{df}{=} (\Sigma_{\mathbf{u}}, \prec_u, \sqsubset_u)^\diamond \qquad S_{\mathbf{u}} \stackrel{df}{=} \left(\Sigma_{\mathbf{u}}, \bigcap_{x \in \mathbf{u}} \triangleleft_x, \bigcap_{x \in \mathbf{u}} \triangleleft_x^\frown \right)$$
■

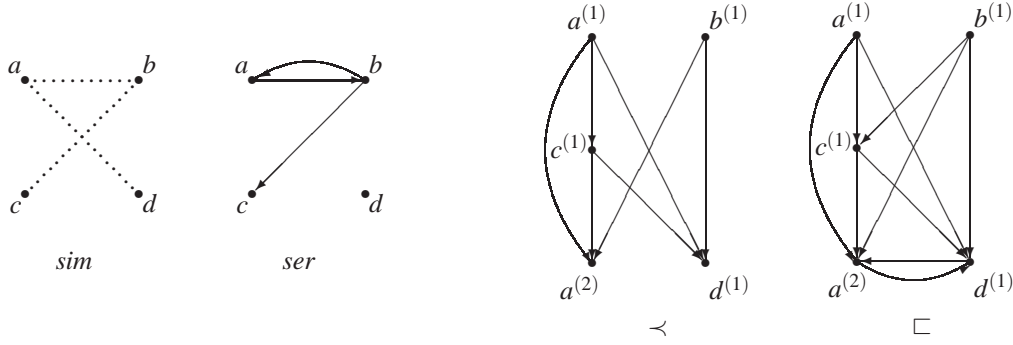


Figure 2: An example of the relations sim , ser on $E = \{a, b, c, d\}$, and the so-structure (X, \prec, \sqsubset) defined by the comtrace $[\{a, b\}\{c\}\{a, d\}]_{\equiv_{ser}} = \{\{a, b\}\{c\}\{a, d\}, \{a\}\{b\}\{c\}\{a, d\}, \{a\}\{b, c\}\{a, d\}, \{b\}\{a\}\{c\}\{a, d\}\}$.

The relational structure $S^{\{u\}}$ is the so-structure induced by the single step sequence u and $S_{\mathbf{u}}$ is the so-structure defined by the comtrace \mathbf{u} . The following theorem justifies the names and summarizes some nontrivial results concerning the so-structures generated by comtraces.

Theorem 10.2 ([12] and [13, Theorem 4.10 and Theorem 4.12]). *For all $u, v \in \mathbb{S}^*$, we have*

1. $S^{\{u\}}$ and $S_{[u]}$ are so-structures,
2. $u \equiv v \iff S^{\{u\}} = S^{\{v\}}$,
3. $S^{\{u\}} = S_{[u]}$,
4. $ext(S_{[u]}) = \{\triangleleft_x \mid x \in [u]\}$. □

In principle, Theorem 10.2 states that the so-structures $S^{\{u\}}$ and $S_{[u]}$ from Definition 10.3 are identical and their set of stratified extensions is exactly the comtrace $[u]$ with step sequences interpreted as stratified orders. However, from an algorithmic point of view, the definition of $S^{\{u\}}$ is much more interesting, since building the relations \prec_u and \sqsubset_u and getting their \Diamond -closure, which in turn can be reduced to computing transitive closure of relations, can be done very efficiently. In contrast, a direct use of the $S_{[u]}$ definition requires precomputing up to exponentially many elements of the comtrace $[u]$.

Figure 2 shows an example of a comtrace and the so-structure it generates.

Theorem 10.2 characterizes so-structures *derived from a given comtrace* and was proved mainly in [12]. However, the reciprocal problem which characterizes the *comtrace derived from a given finite* so-structure has not been formally dealt with so far. We will now study this problem.

Let $S = (X, \prec, \sqsubset)$ be a so-structure. For each stratified order $\triangleleft \in ext(S)$, recall that Ω_{\triangleleft} denotes the step sequence defined by \triangleleft . Note that each element appearing in Ω_{\triangleleft} is unique so enumeration of elements in X are not needed.

We will start with the definitions of relations *simultaneity* and *serializability* that are defined by a given so-structure $S = (X, \prec, \sqsubset)$.

Definition 10.4. For each $a, b \in X$:

1. $(a, b) \in \text{sim}_S \xLeftrightarrow{\text{df}} a \frown_{\prec} b$,
2. $(a, b) \in \text{ser}_S \xLeftrightarrow{\text{df}} a \frown_{\prec} b \wedge \neg(b \sqsubset a)$. ■

The names *simultaneity* and *serializability* are justified by the following result.

Proposition 10.1. For each $a, b \in X$:

1. $(a, b) \in \text{sim}_S \iff \exists \triangleleft \in \text{ext}(S). a \frown_{\triangleleft} b$,
2. $(a, b) \in \text{ser}_S \iff (\exists \triangleleft \in \text{ext}(S). a \frown_{\triangleleft} b) \wedge (\exists \triangleleft \in \text{ext}(S). a \triangleleft b)$,
3. $(a, b) \notin \text{ser}_S \iff a \prec b \vee b \sqsubset a$.

PROOF. 1. This is a consequence of Theorem 4.1 and Theorem 4.2. We have:

$$\begin{aligned}
& (a, b) \in \text{sim}_S \\
& \iff \neg(a \prec b) \wedge \neg(b \prec a) && \langle \text{Definition 10.4} \rangle \\
& \iff \neg(\forall \triangleleft \in \text{ext}(S). a \triangleleft b) \wedge \neg(\forall \triangleleft \in \text{ext}(S). b \triangleleft a) && \langle \text{Theorem 4.1} \rangle \\
& \iff ((\exists \triangleleft \in \text{ext}(S). a \triangleleft b) \wedge (\exists \triangleleft \in \text{ext}(S). b \triangleleft a)) \\
& \quad \vee (\exists \triangleleft \in \text{ext}(S). a \frown_{\triangleleft} b)) \\
& \iff \exists \triangleleft \in \text{ext}(S). a \frown_{\triangleleft} b && \langle \text{Theorem 4.2} \rangle
\end{aligned}$$

2. Follows from (1) and Theorem 4.1.

3. Follows from Definition 10.4, Theorem 4.1 and (2). ■

If stratified orders from $\text{ext}(S)$ are interpreted as observations of concurrent histories (see Section 4, and [9, 11]), then $(a, b) \in \text{sim}_S$ means there is an observation in $\text{ext}(S)$ where a and b are executed simultaneously and $(a, b) \in \text{ser}_S$ means there are *equivalent* observations where in one observation a and b are executed simultaneously, and in another b follows a . Proposition 10.1(3) will often be used in the subsequent proofs.

Since each $\triangleleft \in \text{ext}(S)$ can be interpreted as a step sequence, we define a relational structure induced by \triangleleft similarly to the definition of $S^{\{u\}}$ for a given step sequence u , but this time with great simplification.

Definition 10.5. For each $\triangleleft \in \text{ext}(S)$, define $S^{\{\triangleleft\}} \stackrel{\text{df}}{=} (X, \triangleleft \setminus \text{ser}_S, \triangleleft \frown \setminus \text{ser}_S^{-1})$. ■

It is worth noticing that the relational structure $S^{\{\triangleleft\}}$ is a so-structure induced by \triangleleft , and the relations $\triangleleft \setminus \text{ser}_S, \triangleleft \frown \setminus \text{ser}_S^{-1}$ play the similar roles as \prec_u, \sqsubset_u from Definition 10.2 except the \diamond -closure is *not* needed. The subtle reason behind this simplification is shown in the following proposition.

Proposition 10.2. For every $\triangleleft \in \text{ext}(S)$, we have:

1. $\triangleleft \setminus \text{ser}_S = \prec = \prec \setminus \text{ser}_S$,
2. $\triangleleft \frown \setminus \text{ser}_S^{-1} = \sqsubset = \sqsubset \setminus \text{ser}_S^{-1}$.

$$3. S^{\{\triangleleft\}} = (X, \prec, \sqsubset).$$

PROOF. 1. We now show the first equality.

$$\begin{aligned} & a \triangleleft b \wedge (a, b) \notin ser_S \\ \iff & a \triangleleft b \wedge (a \prec b \vee b \sqsubset a) && \langle \text{Proposition 10.1(3)} \rangle \\ \iff & (a \triangleleft b \wedge a \prec b) \vee (a \triangleleft b \wedge b \sqsubset a) \iff a \prec b \vee False && \langle \text{Theorem 4.1} \rangle \end{aligned}$$

For the second equality, we have $\prec \setminus ser_S = (\triangleleft \setminus ser_S) \setminus ser_S = \triangleleft \setminus ser_S = \prec$.

2. We will show the first equality.

$$\begin{aligned} & a \triangleleft^\cap b \wedge (b, a) \notin ser_S \\ \iff & a \triangleleft^\cap b \wedge (b \prec a \vee a \sqsubset b) && \langle \text{Proposition 10.1(3)} \rangle \\ \iff & (a \triangleleft^\cap b \wedge b \prec a) \vee (a \triangleleft^\cap b \wedge a \sqsubset b) \iff False \vee a \sqsubset b && \langle \text{Theorem 4.1} \rangle \end{aligned}$$

The second equality immediately follows.

3. Follows from (1) and (2). \square

We have just shown that every so-structure $S = (X, \prec, \sqsubset)$ is equal to $S^{\{\triangleleft\}}$ for any $\triangleleft \in ext(S)$. Note that this proof does not assume that S is finite. Hence, Proposition 10.2 also holds when S is an *infinite* so-structure. This proposition can be interpreted as a generalization of the following folklore result on recovering a partial order from any of its total extension (by replacing partial orders with so-structures and total extensions with stratified extensions).

Proposition 10.3. *For every partial order $<$ and every total order $\leq \in Total(<)$, $\leq \setminus \prec_\leq = <$.*

$$\text{PROOF. } a(\leq \setminus \prec_\leq)b \iff (a \leq b \wedge (a < b \vee b < a)) \iff a < b. \quad \square$$

We will end this section by proving that if S is a finite so-structure, then the set $ext(S)$, when interpreted as a set of step sequences, is truly a comtrace over the comtrace alphabet (X, sim_S, ser_S) . Moreover, the so-structure generated by this particular comtrace is exactly S .

Definition 10.6. For every finite so-structure $S = (X, \prec, \sqsubset)$, we define:

1. $\Theta_S = (X, sim_S, ser_S)$,
2. $\mathfrak{C}(S) = \{\Omega_{\triangleleft} \mid \triangleleft \in ext(S)\}$. \blacksquare

Note that since sim_S and ser_S clearly satisfy the properties from Definition 5.1, the triple Θ_S is a *comtrace alphabet*. So we can define the comtrace congruence \equiv_{ser_S} with respect to Θ_S . We will call $\mathfrak{C}(S)$ the *comtrace generated by $S = (X, \prec, \sqsubset)$* . Theorem 10.3, the main result of this section will justify this name.

Theorem 10.3. *Let $S = (X, \prec, \sqsubset)$ be a finite so-structure. For every $\triangleleft \in ext(S)$, we have:*

1. $S_{[\Omega_{\triangleleft}]} = S^{\{\Omega_{\triangleleft}\}} = S^{\{\triangleleft\}} = S$,
2. $\mathfrak{C}(S) = [\Omega_{\triangleleft}]$.

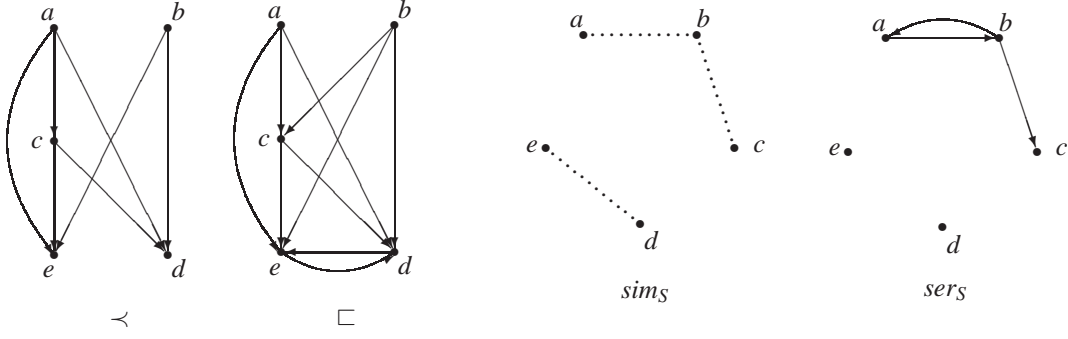


Figure 3: An example of a so-structure $S = (X, \prec, \sqsubset)$, where $X = \{a, b, c, d, e\}$, and the relations sim_S, ser_S . The so-structure S , defines the comtrace $[\{a, b\}\{c\}\{d, e\}]_{\equiv_{ser_S}} = \{\{a, b\}\{c\}\{d, e\}, \{a\}\{b\}\{c\}\{e, d\}, \{a\}\{b, c\}\{e, d\}, \{b\}\{a\}\{c\}\{e, d\}\}$.

PROOF. 1. First we need to check that Ω_{\triangleleft} is indeed a step sequence over θ . Assume that $\Omega_{\triangleleft} = A_1 \dots A_n$, then since $\triangleleft \in ext(G)$, for each A_i ($1 \leq i \leq n$), we have if $a, b \in A_i$ and $a \neq b$, then $a \triangleleft b$. Hence, by Definition 10.4, we have $(a, b) \in sim$. Since Ω_{\triangleleft} is a step sequence over θ , we can construct $S^{\{\Omega_{\triangleleft}\}} = (X, \triangleleft \setminus ser_S, \triangleleft \cap \setminus ser_S^{-1})^\diamond$ as from Definition 10.3. Thus, we have:

$$\begin{aligned}
 S_{[\Omega_{\triangleleft}]} &= S^{\{\Omega_{\triangleleft}\}} && \langle \text{Theorem 10.2(3)} \rangle \\
 &= (X, \triangleleft \setminus ser_S, \triangleleft \cap \setminus ser_S^{-1})^\diamond && \langle \text{Definition 10.3} \rangle \\
 &= (X, \prec, \sqsubset)^\diamond && \langle \text{Proposition 10.2} \rangle \\
 &= (X, \prec, \sqsubset) = S && \langle \text{Theorem 10.1(4)} \rangle \\
 &= S^{\{\triangleleft\}} && \langle \text{Proposition 10.2} \rangle
 \end{aligned}$$

2. From (1) and Theorem 10.2(2,3), we have $ext(S_{[\Omega_{\triangleleft}]}) = ext(S) = \{\triangleleft_u \mid u \in [\Omega_{\triangleleft}]\}$. But this implies that $[\Omega_{\triangleleft}] = \{\Omega_{\blacktriangleleft} \mid \blacktriangleleft \in ext(S)\} = \mathfrak{C}(S)$ by Definition 10.6. \square

Theorem 10.3(2) states that for each so-structure $S = (X, \prec, \sqsubset)$, the set of all of its stratified extensions $ext(S)$, when interpreted as a set of step sequences is in fact a comtrace over the comtrace alphabet $\Theta_S = (X, sim_S, ser_S)$, so we can call $\mathfrak{C}(S)$ the *comtrace generated by S*. The equality $S_{[\Omega_{\triangleleft}]} = S$ in Theorem 10.3(1) says that the so-structure defined by the comtrace $\mathfrak{C}(S)$ is exactly S , so comtraces and finite so-structures can be interpreted as equivalent or *tantamount* (cf. [9]) concurrent models.

An example of a so-structure and the comtrace it generates is presented in Figure 3.

11. Generalized Stratified Order Structures Generated by Generalized Comtraces

The relationship between g-comtraces and gso-structures is in principle the same as the relationship between comtraces and so-structures discussed in the previous section. Each g-comtrace uniquely determines a finite gso-structure and each finite gso-structure can be represented by a

g-comtrace. However the proofs and even formulations of those results are much longer and more complex than in the case of the relationship between comtraces and so-structures. The difficulties stem mainly from the following facts:

- The definition of gso-structures is implicit, it involves using the induced so-structures (see Definition 4.3), which makes practically all definitions much more complex (especially the counterpart of \diamond -closure), and the use of Theorem 4.3 more difficult than the use of Theorem 4.1.
- The internal property expressed by Theorem 4.2, which says that $ext(S)$ conforming to paradigm π_3 of [11], does not hold for gso-structures.
- g-Comtraces do not have a ‘natural’ canonical form with a well understood interpretation.
- The relation inl introduces plenty of irregularity and substantially increases the numbers of cases that need to be considered in many proofs.

In this chapter, we will prove the analogue of Theorem 10.2 showing that every g-comtrace uniquely determines a finite gso-structure.

11.1. Commutative closure of relational structures

We will start with the notion of *commutative closure* of a relational structure. It is an extension of the concept of \diamond -closure (see Definition 10.1) which was used in [12] and the previous section to construct finite so-structures from single step sequences or stratified orders.

Definition 11.1 (Commutative Closure).

Let $G = (X, R_1, R_2)$ be any relational structure, and let $R_3 = R_1 \cap R_2^*$. Using the notation from Definition 10.1, the *commutative closure* of the relational structure G is defined as

$$G^{\boxtimes} = (X, (\prec_{R_3 R_2})^{\text{sym}} \cup R_1, \sqsubseteq_{R_3 R_2}).$$

■

The motivation behind the above definition is similar to that for \diamond -closure: for ‘reasonable’ R_1 and R_2 , $(X, R_1, R_2)^{\boxtimes}$ should be a gso-structure. Intuitively the \boxtimes -closure is also a generalization of transitive closure for relations. Note that if $R_1 = R_2$ then $(X, R_1, R_2)^{\boxtimes} = (X, (R_1^+)^{\text{sym}}, R_1^+)$. Since the definition of gso-structures involves the definition of so-structures (see Definition 4.3), the definition of \boxtimes -closure uses the concept of \diamond -closure.

Note that we do not have an equivalence of Theorem 10.1 for \boxtimes -closure. The reason is that \boxtimes -closure is tailored to simplify the proofs that we will show in the next section rather than to be a closure operator by itself. Nevertheless, \boxtimes -closure does have some general properties which are extremely useful in our proofs.

The first property shows that the relationship between \boxtimes -closure and \diamond -closure corresponds to the relationship between gso-structures and so-structures as stated in Definition 4.3.

Proposition 11.1. *Let (X, R_1, R_2) be a relational structure and $R_3 = R_1 \cap R_2^*$. If $(X, \prec_0, \sqsubseteq_0) = (X, R_3, R_2)^{\diamond}$ is a so-structure, then $\prec_0 = (\prec_0^{\text{sym}} \cup R_2) \cap \sqsubseteq_0$.*

PROOF. (\subseteq) Since $(X, \prec_0, \sqsubseteq_0) = (X, R_3, R_2)^{\diamond}$, by definition of \diamond -closure, $\prec_0 \subseteq \sqsubseteq_0$. Since we also have $\prec_0 \subseteq (\prec_0 \cup R_2)$, it follows that $\prec_0 \subseteq (\prec_0^{\text{sym}} \cup R_2) \cap \sqsubseteq_0$.

(\supseteq) Suppose that $(x, y) \in (\prec_0^{\text{sym}} \cup R_2) \cap \sqsubseteq_0$ and $\neg(x \prec_0 y)$. There are two cases to consider:

- (a) $y \prec_0 x$ and $x \sqsubset_0 y$: Since $(X, \prec_0, \sqsubset_0)$ is a so-structure, we have $y \prec_0 x \implies \neg(x \sqsubset_0 y)$, a contradiction.
- (b) $(x, y) \in R_2$ and $x \sqsubset_0 y$: Since $(X, \prec_0, \sqsubset_0) = (X, R_3, R_2)^\diamond$, we have $\prec_0 = (R_3 \cup R_2)^* \circ R_3 \circ (R_3 \cup R_2)^*$ and $\sqsubset_0 = (R_3 \cup R_2)^* \setminus id_X$. Since $x \sqsubset_0 y$ and $\neg(x \prec_0 y)$, we have $(x, y) \in (R_2^* \setminus id_X)$. Since $(x, y) \in (R_2^* \setminus id_X)$ and $(x, y) \in R_1$, we have xR_3y . Hence, $x \prec_0 y$, a contradiction. \square

The second property we will show is the monotonicity of \bowtie -closure.

Proposition 11.2. *If $G_1 = (X, R_1, R_2)$ and $G_2 = (X, Q_1, Q_2)$ are two relational structures such that $G_1 \subseteq G_2$, then $G_1^\bowtie \subseteq G_2^\bowtie$.*

PROOF. Let $R_3 = R_1 \cap R_2^*$ and $Q_3 = Q_1 \cap Q_2^*$. Since $R_1 \subseteq Q_1$ and $R_2 \subseteq Q_2$, we have $R_3 \subseteq Q_3$, and $(X, R_3, R_2)^\diamond \subseteq (X, Q_3, Q_2)^\diamond$, i.e., $\prec_{R_3R_2} \subseteq \prec_{Q_3Q_2}$ and $\sqsubset_{R_3R_2} \subseteq \sqsubset_{Q_3Q_2}$. This immediately implies $G_1^\bowtie \subseteq G_2^\bowtie$. \square

Another desirable and very useful property of \bowtie -closure is that gso-structures are fixed points of \bowtie -closure.

Proposition 11.3. *If $G = (X, \diamond, \sqsubset)$ is a gso-structure then $G = G^\bowtie$.*

PROOF. Since G is a gso-structure, by Definition 4.3, $S_G = (X, \prec_G, \sqsubset)$ is a so-structure. Hence, by Theorem 10.1(4), $S_G = S_G^\diamond$, which implies $\sqsubset = (\prec_G \cup \sqsubset)^* \setminus id_X$. But since S_G is a so-structure, $\prec_G \subseteq \sqsubset$. So $\sqsubset = \sqsubset^* \setminus id_X$. Let $\prec = \diamond \cap \sqsubset^*$. Then since \diamond is irreflexive,

$$\prec = \diamond \cap \sqsubset^* = \diamond \cap (\sqsubset^* \setminus id_X) = \diamond \cap \sqsubset = \prec_G.$$

Hence, $(X, \prec, \sqsubset) = (X, \prec_G, \sqsubset)$ is a so-structure. By Theorem 10.1(4), we know $(X, \prec, \sqsubset) = (X, \prec, \sqsubset)^\diamond$. So from Definition 11.1, $G^\bowtie = (X, \prec^{\text{sym}} \cup \diamond, \sqsubset)$. Since \diamond is symmetric and $\prec \subseteq \diamond$, we have $\prec^{\text{sym}} \cup \diamond = \diamond$. Thus, $G = G^\bowtie$. \square

The remaining properties of \bowtie -closure have to be proved when needed for specific relations R_1 and R_2 .

11.2. Generalized Stratified Order Structure Generated by a Step Sequence

We will now introduce a construction that derives a gso-structure from a single step sequence over a given g-comtrace alphabet. The idea of the construction is the same as $S^{\{u\}}$ from the previous section. First we construct some relational invariants and next we will use \bowtie -closure in the similar manner as \diamond -closure was used for $S^{\{u\}}$. However the construction will be more elaborate and will require full use of the notation from Section 2.3 that allows us to define the formal relationship between step sequences and (labelled) stratified orders. We will also need the following two useful operators for relations.

Definition 11.2. Let R be a binary relation on X . We define the

- symmetric intersection of R as $R^\bowtie \stackrel{\text{df}}{=} R \cap R^{-1}$, and
- the complement of R as $R^C \stackrel{\text{df}}{=} (X \times X) \setminus R$. ■

Let $\Theta = (E, \text{sim}, \text{ser}, \text{inl})$ be a g-comtrace alphabet. Note that if $u \equiv w$ then $\Sigma_u = \Sigma_w$ so for every g-comtrace $s = [s] \in \mathbb{S}^* / \equiv$, we can define $\Sigma_s = \Sigma_s$.

Definition 11.3. Given a step sequence $s \in \mathbb{S}^*$.

1. Let the relations $\diamond_s, \sqsubseteq_s, \prec_s \subseteq \Sigma_s \times \Sigma_s$ be defined as follows:

$$\alpha \diamond_s \beta \stackrel{\text{df}}{\iff} (l(\alpha), l(\beta)) \in \text{inl} \quad (11.1)$$

$$\alpha \sqsubseteq_s \beta \stackrel{\text{df}}{\iff} \alpha \prec_s \beta \wedge (l(\beta), l(\alpha)) \notin \text{ser} \cup \text{inl} \quad (11.2)$$

$$\alpha \prec_s \beta \stackrel{\text{df}}{\iff} \alpha \prec_s \beta \wedge \left(\begin{array}{l} (l(\alpha), l(\beta)) \notin \text{ser} \cup \text{inl} \\ \vee (\alpha, \beta) \in \diamond_s \cap ((\sqsubseteq_s^*)^{\text{m}} \circ \diamond_s^{\text{c}} \circ (\sqsubseteq_s^*)^{\text{m}}) \\ \vee \left(\begin{array}{l} (l(\alpha), l(\beta)) \in \text{ser} \\ \wedge \exists \delta, \gamma \in \Sigma_s. \left(\begin{array}{l} \delta \prec_s \gamma \wedge (l(\delta), l(\gamma)) \notin \text{ser} \\ \alpha \sqsubseteq_s^* \delta \sqsubseteq_s^* \beta \wedge \alpha \sqsubseteq_s^* \gamma \sqsubseteq_s^* \beta \end{array} \right) \end{array} \right) \end{array} \right) \quad (11.3)$$

2. The triple

$$G^{\{s\}} \stackrel{\text{df}}{=} (\Sigma_s, \prec_s \cup \diamond_s, \prec_s \cup \sqsubseteq_s)^{\boxtimes}$$

is called the *relational structure induced by the step sequence s*. ■

The intuition of Definition 11.3 is similar to that of Definition 10.2. Given a step sequence s and the g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$, without analyzing any other elements of $[s]$ except s itself, we would like to be able to construct the same gso-structure as the one defined by the whole g-comtrace. So we will define appropriate “local” invariants $\diamond_s, \sqsubseteq_s$ and \prec_s from the sequence s .

- (a) Equation 11.1 is used to construct the relationship \diamond_s , where two event occurrences α and β might possibly be commutative because they are related by the *inl* relation.
- (b) Equation 11.2 define the not later than relationship and this happens when α occurs not later than β on the step sequence s and $\{\alpha, \beta\}$ cannot be serialized into $\{\beta\}\{\alpha\}$, and α and β are not commutative.
- (c) Equation 11.3 is the most complicated one, since we want to take into consideration the “earlier than” relationships which are not taken care of by the commutative closure. There are three such cases:
 - (i) α occurs before β on the step sequence s , and two event occurrences α and β cannot be put together into a single step ($(\alpha, \beta) \notin \text{ser}$) and are not commutative ($(\alpha, \beta) \notin \text{inl}$).
 - (ii) α and β are supposed to be commutative but they can be flipped into β and α because α is “synchronous” with some γ and β is “synchronous” with some δ , and (γ, δ) is not in *inl* (“synchronous” in a sense that they must happen simultaneously).
 - (iii) (α, β) is in *ser* but they can never be put together into a single step because there are some distinct event occurrences γ and δ which are squeezed between α and β (always occur between α and δ) such that δ occurs before γ and (δ, γ) is not in *ser* (δ and γ will never be put together into a single step).

After building all of these “local” invariants from the step sequence s , all of other “global” invariants which can be inferred from the axioms of the gso-structure definition are fully constructed by the commutative closure.

The next proposition will show that the relations from $G^{\{s\}}$ really corresponds to positional invariants of all the step sequences from the g-comtrace $[s]$.

Proposition 11.4. *Let $s \in \mathbb{S}^*$, $G^{\{s\}} = (\Sigma_s, \diamond, \sqsubset)$, and $\prec = \diamond \cap \sqsubset$. If $\alpha, \beta \in \Sigma_s$, then*

1. $\alpha \diamond \beta \iff \forall u \in [s]. \text{pos}_u(\alpha) \neq \text{pos}_u(\beta)$
2. $\alpha \sqsubset \beta \iff \alpha \neq \beta \wedge \forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta)$
3. $\alpha \prec \beta \iff \forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)$
4. *If $l(\alpha) = l(\beta)$ and $\text{pos}_s(\alpha) < \text{pos}_s(\beta)$, then $\alpha \prec \beta$.* □

Eventhough the results of the above proposition are expected and look deceptively simple, the proof is long and highly technical and can be found in Appendix A. We will next show that $G^{\{s\}}$ is indeed a gso-structure.

Theorem 11.1. *Let $s \in \mathbb{S}^*$. Then $G^{\{s\}} = (\Sigma_s, \diamond, \sqsubset)$ is a gso-structure.*

PROOF. Since $\diamond = \bigcap_{u \in [s]} \triangleleft_u^{\text{sym}}$ and $\triangleleft_u^{\text{sym}}$ is irreflexive and symmetric, \diamond is irreflexive and symmetric. Since $\sqsubset = \bigcap_{u \in [s]} \triangleleft_u$ and \triangleleft_u is irreflexive, \sqsubset is irreflexive.

Let $\prec = \diamond \cap \sqsubset$, it remains to show that $S = (\Sigma, \prec, \sqsubset)$ satisfies the conditions S1–S4 of Definition 4.1. Since \sqsubset is irreflexive, S1 is satisfied. Since $\prec \subseteq \sqsubset$, S2 is satisfied. Assume $\alpha \sqsubset \beta \sqsubset \gamma$ and $\alpha \neq \gamma$. Then

$$\begin{aligned}
 & \alpha \sqsubset \beta \sqsubset \gamma \wedge \alpha \neq \gamma \\
 \implies & (\alpha, \beta) \in \bigcap_{u \in [s]} \triangleleft_u \wedge (\beta, \gamma) \in \bigcap_{u \in [s]} \triangleleft_u \wedge \alpha \neq \gamma && \langle \text{Theorem 11.2} \rangle \\
 \implies & \forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta) \leq \text{pos}_u(\gamma) \wedge \alpha \neq \gamma && \langle \text{Definition of } \triangleleft_u \rangle \\
 \implies & \alpha \sqsubset \gamma && \langle \text{Proposition 11.4(2)} \rangle
 \end{aligned}$$

Hence, S3 is satisfied. Next we assume that $\alpha \prec \beta \sqsubset_s \gamma$. Then

$$\begin{aligned}
 & \alpha \prec \beta \sqsubset \gamma \\
 \implies & (\alpha, \beta) \in \bigcap_{u \in [s]} (\triangleleft_u \cap \triangleleft_u^{\text{sym}}) \wedge (\beta, \gamma) \in \bigcap_{u \in [s]} (\triangleleft_u \cap \triangleleft_u^{\text{sym}}) && \langle \text{Theorem 11.2} \rangle \\
 \implies & (\forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta) \wedge \text{pos}_u(\alpha) \neq \text{pos}_u(\beta)) \\
 & \quad \wedge (\forall u \in [s]. \text{pos}_u(\beta) \leq \text{pos}_u(\gamma) \wedge \text{pos}_u(\beta) \neq \text{pos}_u(\gamma)) && \langle \text{Definition of } \triangleleft_u \rangle \\
 \implies & \forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\gamma) \\
 \implies & \alpha \prec \gamma && \langle \text{Proposition 11.4(3)} \rangle
 \end{aligned}$$

Similarly, we can show $\alpha \sqsubset \beta \prec \gamma \implies \alpha \prec \gamma$. Thus, S4 is satisfied. □

Theorem 11.2 justifies the following definition:

Definition 11.4. For every step sequence s , $G^{\{s\}} = (\Sigma_s, \prec_s \cup \diamond_s, \prec_s \cup \sqsubset_s)^{\boxtimes}$ is the gso-structure induced by s . ■

Note that Proposition 11.4 also implies that we can construct the gso-structure $G^{\{s\}}$ if all the step sequences of a g-comtrace are known. We will first show how to define the gso-structure induced from all the positional invariants of all the step sequences of a g-comtrace.

Definition 11.5. For every $s \in \mathbb{S}^*/\equiv$, we define $G_s = \left(\Sigma_s, \bigcap_{u \in s} \triangleleft_u^{\text{sym}}, \bigcap_{u \in s} \triangleleft_u^\wedge \right)$. ■

We will now show that given a step sequence s over a g-comtrace alphabet, the definition of $G^{\{s\}}$ and the definition of $G_{[s]}$ yield exactly the same gso-structure.

Theorem 11.2. Let $s \in \mathbb{S}^*$. Then $G^{\{s\}} = G_{[s]}$.

PROOF. Let $G^{\{s\}} = (\Sigma_s, \triangleleft, \sqsubset)$ and $\alpha, \beta \in \Sigma_s$. Then by Proposition 11.4(1, 2), we have

$$\begin{aligned} \alpha \triangleleft \beta &\iff \forall u \in [s]. \text{pos}_u(\alpha) \neq \text{pos}_u(\beta) \iff (\alpha, \beta) \in \bigcap_{u \in [s]} \triangleleft_u^{\text{sym}} \\ \alpha \sqsubset \beta &\iff (\alpha \neq \beta \wedge \forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta)) \iff (\alpha, \beta) \in \bigcap_{u \in [s]} (\triangleleft_u^\wedge)^{\text{sym}} \end{aligned}$$

Hence, $G^{\{s\}} = (\Sigma_s, \triangleleft, \sqsubset) = \left(\Sigma_s, \bigcap_{u \in [s]} \triangleleft_u^{\text{sym}}, \bigcap_{u \in [s]} \triangleleft_u^\wedge \right) = G_{[s]}$. □

At this point it is worth discussing the roles of the two different definitions of the gso-structures generated from a given g-comtrace. Definition 11.3 allows us to build the gso-structure by looking at a single step sequence of the g-comtrace and its g-comtrace alphabet. On the other hand, to build the gso-structure from a g-comtrace using Definition 11.5, we need to know either all the positional invariants or all elements of the g-comtrace. By Theorem 11.2, these two definitions are equivalent. In our proof, Definition 11.3 is more convenient when we want to deduce the properties of the gso-structure from a single step sequence and a g-comtrace alphabet. Definition 11.3 will be used to reconstruct the relations of a gso-structure when some positional invariants of a g-comtrace are known.

11.3. Generalized Stratified Order Structures Generated by Generalized Comtraces

In this section, we want to show that the construction from Definition 11.3 indeed yields a gso-structure representation of comtraces. But before doing so, we need some preliminary results.

Proposition 11.5. Let $s \in \mathbb{S}^*$. Then $\triangleleft_s \in \text{ext}(G^{\{s\}})$.

PROOF. Let $G^{\{s\}} = (\Sigma, \triangleleft, \sqsubset)$. By Proposition 11.4, for all $\alpha, \beta \in \Sigma$,

$$\begin{aligned} \alpha \triangleleft \beta &\implies \text{pos}_s(\alpha) \neq \text{pos}_s(\beta) \implies \alpha \triangleleft_s \beta \vee \beta \triangleleft_s \alpha \implies \alpha \triangleleft_s^{\text{sym}} \beta \\ \alpha \sqsubset \beta &\implies \text{pos}_s(\alpha) \leq \text{pos}_s(\beta) \implies \alpha \triangleleft_s^\wedge \beta \end{aligned}$$

Hence, by Definition 4.4, we get $\triangleleft_s \in \text{ext}(G^{\{s\}})$. □

Proposition 11.6.

Let $s \in \mathbb{S}^*$. If $\triangleleft \in \text{ext}(G^{\{s\}})$, then there is a step sequence $u \in \mathbb{S}^*$ such that $\triangleleft = \triangleleft_u$.

PROOF. Let $G^{\{s\}} = (\Sigma_s, \diamond, \sqsubseteq)$ and $\Omega_{\triangleleft} = B_1 \dots B_k$. We will show that $u = l[B_1] \dots l[B_k]$ is a step sequence such that $\triangleleft = \triangleleft_u$.

Suppose $\alpha, \beta \in B_i$ are two distinct event occurrences such that $(l(\alpha), l(\beta)) \notin \text{sim}$. Then $\text{pos}_s(\alpha) \neq \text{pos}_s(\beta)$, which by Proposition 11.4 implies that $\alpha \diamond \beta$. Since $\triangleleft \in \text{ext}(G^{\{s\}})$, by Definition 4.4, $\alpha \triangleleft \beta$ or $\beta \triangleleft \alpha$ contradicting that $\alpha, \beta \in B_i$. Thus, we have shown for all B_i ($1 \leq i \leq k$),

$$\alpha, \beta \in B_i \wedge \alpha \neq \beta \implies (l(\alpha), l(\beta)) \notin \text{sim} \quad (11.4)$$

By Proposition A.1(2), if $e^{(i)}, e^{(j)} \in \Sigma_s$ and $i \neq j$ then $\forall u \in [s]. \text{pos}_u(e^{(i)}) \neq \text{pos}_u(e^{(j)})$. So it follows from Proposition 11.4(1) that $e^{(i)} \diamond e^{(j)}$. Since $\triangleleft \in \text{ext}(G^{\{s\}})$, by Definition 4.4,

$$\text{If } e^{(k_0)} \in B_k \text{ and } e^{(m_0)} \in B_m, \text{ then } k_0 \neq m_0 \iff k \neq m \quad (11.5)$$

From (11.4) it follows that u is a step sequence over θ . Also by (11.5), $\text{pos}_u^{-1}[\{i\}] = B_i$ and $|l[B_i]| = |B_i|$ for all i . Hence, $\Omega_{\triangleleft} = \Omega_{\triangleleft_u}$, which implies $\triangleleft = \triangleleft_u$. \square

We want to show that two step sequences over the same g-comtrace alphabet induce the same gso-structure if and only if they belong to the same g-comtrace (Theorem 11.3 below). The proof of an analogous result for comtraces from [12] is simpler because every comtrace has a unique natural canonical representation that is both greedy and maximally concurrent and can be easily constructed. Moreover the canonical representation for comtraces correspond to the unique greedy stratified extension of appropriate causality relation \prec (see [12]). Nothing similar holds for g-comtraces. For g-comtraces both natural representations, GMC and MC, are not unique. The g-canonical representation (Definition 8.4) is unique but its uniqueness is artificial, it is induced by some total lexicographical order $<^{lex}$ imposed on step sequences (Definition 8.3); from all GMC-representations we just have to pick the one that is lexicographically smallest. Nevertheless this lexicographical order $<^{lex}$ will be one of the basic tools used in this subsection. However the lack of natural unique representation will make our reasoning much more difficult.

Lemma 11.1. *Let s be a step sequence over a g-comtrace alphabet $(E, \text{ser}, \text{sim}, \text{inl})$ and $<_E$ be any total order on E . Let $u = A_1 \dots A_n$ be the g-canonical representation of $[s]$ (i.e., u is the least element of the g-comtrace $[s]$ w.r.t. $<^{lex}$). Let $G^{\{s\}} = (\Sigma, \diamond, \sqsubseteq)$ and $\triangleleft = \diamond \cap \sqsubseteq$. Let $\text{mins}_{\triangleleft}(X)$ denote the set of all minimal elements of X w.r.t. \triangleleft and define*

$$Z(X) \stackrel{\text{df}}{=} \left\{ Y \subseteq \text{mins}_{\triangleleft}(X) \mid (\forall \alpha, \beta \in Y. \neg(\alpha \diamond \beta)) \wedge (\forall \alpha \in Y. \forall \beta \in X \setminus Y. \neg(\beta \sqsubseteq \alpha)) \right\}$$

Let $\bar{u} = \bar{A}_1 \dots \bar{A}_n$ be the enumerated step sequence of u . Then A_i is the least element of the set $\{l[Y] \mid Y \in Z(\Sigma \setminus \biguplus(\bar{A}_1 \dots \bar{A}_{i-1}))\}$ w.r.t. $<^{st}$. \square

Before presenting the proof, we will explain the intuitions behind the definition of the set $Z(X)$. Let us consider $Z(\Sigma)$ first. We want A_1 to be the least element of the set $\{l[Y] \mid Y \in Z(X)\}$. In short, we want to construct A_1 by looking only at the gso-structure G without having to construct up to exponentially many stratified extensions of G . The most difficult part of this proof is to show that A_1 must be a subset of the set of all minimal elements of (Σ, \triangleleft) satisfying all the constraints of $Z(\Sigma)$, i.e., each $Y \in Z(\Sigma)$ satisfies:

- i. no two elements in Y are commutative,
- ii. for an element $\alpha \in Y$ and $\beta \in \Sigma \setminus Y$, it is not the case that β is not later than α .

Note that we actually define $Z(X)$ instead of $Z(\Sigma)$, because we want to apply it successively to build *all* the steps of the g-canonical representation of $G^{\{s\}}$. This lemma can be seen as an algorithm to build the g-canonical representation of $[s]$ by looking only at $G^{\{s\}}$.

PROOF (PROOF OF LEMMA 11.1). We first notice that by Proposition 11.4(4), if $e^{(i)}, e^{(j)} \in \Sigma$ and $i < j$ then $e^{(i)} \prec e^{(j)}$. Hence, for all $\alpha, \beta \in \text{mins}_{\prec}(X)$, where $X \subseteq \Sigma$, we have $l(\alpha) \neq l(\beta)$. This ensures that if $Y \in Z(X)$ and $X \subseteq \Sigma$, then $|Y| = |l[Y]|$.

For all $\alpha \in \overline{A_1}$ and $\beta \in \Sigma$, $\text{pos}_s(\beta) \geq \text{pos}_s(\alpha)$. Hence, by Proposition 11.4(3), $\neg(\beta \prec \alpha)$. Thus,

$$\overline{A_1} \subseteq \text{mins}_{\prec}(X) \quad (11.6)$$

For all $\alpha, \beta \in \overline{A_1}$, since $\text{pos}_s(\beta) = \text{pos}_s(\alpha)$, by Proposition 11.4(1), we have

$$\neg(\alpha \diamond \beta) \quad (11.7)$$

For any $\alpha \in \overline{A_1}$ and $\beta \in \Sigma \setminus \overline{A_1}$, since $\text{pos}_s(\beta) < \text{pos}_s(\alpha)$, by Proposition 11.4(2),

$$\neg(\beta \sqsubset \alpha) \quad (11.8)$$

From (11.6), (11.7) and (11.8), we know that $\overline{A_1} \in Z(\Sigma)$. Hence, $Z(\Sigma) \neq \emptyset$. This ensures the least element of $\{l[Y] \mid Y \in Z(\Sigma)\}$ w.r.t. $<^{st}$ is well defined.

Let $Y_0 \in Z(\Sigma)$ such that $B_0 = l[Y_0]$ be the least element of $\{l[Y] \mid Y \in Z(\Sigma)\}$ w.r.t. $<^{st}$. We want to show that $A_1 = B_0$. Since $<^{st}$ is a total order, we know that $A_1 <^{st} B_0$ or $B_0 <^{st} A_1$ or $A_1 = B_0$. But since $\overline{A_1} \in Z(\Sigma)$ and B_0 be the least element of the set $\{l[B] \mid B \in Z(\Sigma)\}$, $\neg(A_1 <^{st} B_0)$. Hence, to show that $A_1 = B_0$, it suffices to show $\neg(B_0 <^{st} A_1)$.

Suppose that $B_0 <^{st} A_1$. We first want to show that for every nonempty $W \subseteq Y_0$ there is an enumerated step sequence v such that

$$\overline{v} = W_0 \overline{v_0} \equiv \overline{A_1} \dots \overline{A_n} \text{ and } W \subseteq W_0 \subseteq Y_0 \quad (11.9)$$

We will prove this by induction on $|W|$.

Base case. When $|W| = 1$, we let $\{\alpha_0\} = W$. We choose $\overline{v_1} = \overline{E_0} \dots \overline{E_k} \overline{y_1} \equiv \overline{A_1} \dots \overline{A_n}$ and $\alpha_0 \in \overline{E_k}$ ($k \geq 0$) such that for all $\overline{v'} = \overline{E'_0} \dots \overline{E'_{k'}} \overline{y'_1} \equiv \overline{A_1} \dots \overline{A_n}$ and $\alpha_0 \in \overline{E'_{k'}}$, we have

- (i) $\text{weight}(\overline{E_0} \dots \overline{E_k}) \leq \text{weight}(\overline{E'_0} \dots \overline{E'_{k'}})$, and
- (ii) $\text{weight}(\overline{E_{k-1}} \overline{E_k}) \leq \text{weight}(\overline{E'_{k'-1}} \overline{E'_{k'}})$.

We then consider only $\overline{w} = \overline{E_0} \dots \overline{E_k}$. We observe by the way we chose $\overline{v_1}$, we have $\forall \beta \in \uplus(\overline{w})$. ($\beta \neq \alpha_0 \implies \forall t \in [w]. \text{pos}_t(\beta) \leq \text{pos}_t(\alpha_0)$). Hence, since $\overline{w} = \overline{u} \div_R \overline{v_0}$, it follows from Proposition 7.3(1, 2) that

$$\forall \beta \in \uplus(\overline{w}). (\beta \neq \alpha_0 \implies \forall t \in [A_1 \dots A_n]. \text{pos}_t(\beta) \leq \text{pos}_t(\alpha_0))$$

Then it follows from Proposition 11.4(2) that $\forall \beta \in \uplus(\overline{w})$. ($\beta \neq \alpha_0 \implies \beta \sqsubset \alpha_0$). But by the way Y_0 was chosen, we know that $\forall \alpha \in Y_0$. $\forall \beta \in \Sigma \setminus Y_0$. $\neg(\beta \sqsubset \alpha)$. Hence,

$$\uplus(\overline{w}) = (\overline{E_0} \cup \dots \cup \overline{E_k}) \subseteq Y_0 \quad (11.10)$$

We next want to show

$$\forall \alpha \in \overline{E_i}. \forall \beta \in \overline{E_j}. \{\alpha\}\{\beta\} \equiv \{\alpha, \beta\} \quad (0 \leq i < j \leq k) \quad (11.11)$$

Suppose not. Then either $[\{\alpha\}\{\beta\}] = \{\{\alpha\}\{\beta\}\}$ or $[\{\alpha\}\{\beta\}] = \{\{\alpha\}\{\beta\}, \{\beta\}\{\alpha\}\}$. In either case, we have $\forall t \in [\{l(\alpha)\}\{l(\beta)\}]. \text{pos}_t(\alpha) \neq \text{pos}_t(\beta)$. Since $\{\alpha\}\{\beta\} \equiv \pi_{\{\alpha,\beta\}}(\bar{u})$, by Proposition 7.3(3), $\forall t \in [u]. \text{pos}_t(\alpha) \neq \text{pos}_t(\beta)$. So by Proposition 11.4, $\alpha \diamond \beta$. This contradicts that $Y_0 \in Z(\Sigma)$ and $\alpha, \beta \in \Sigma(\bar{w}) \subseteq Y_0$. Thus, we have shown (11.11), which implies that for all $\alpha \in \bar{E}_i$ and $\beta \in \bar{E}_j$ ($0 \leq i < j \leq k$), $(l(\alpha), l(\beta)) \in \text{ser}$. Then $\bar{E}_0 \dots \bar{E}_k \equiv \bigcup_{i=0}^k \bar{E}_i$. Hence, by (11.10) and (11.11), there exists a step sequence v''_1 such that $\bar{v}_1'' = \left(\bigcup_{i=0}^k \bar{E}_i\right) \bar{v}_1 \equiv \bar{A}_1 \dots \bar{A}_n$ and $\{\alpha_0\} \subseteq \bigcup_{i=0}^k \bar{E}_i \subseteq Y_0$.

Inductive step. When $|W| > 1$, we pick an element $\beta_0 \in W$. By applying the induction hypothesis on $W \setminus \{\beta_0\}$, we get a step sequence v_2 such that $\bar{v}_2 = \bar{F}_0 \bar{v}_2 \equiv \bar{A}_1 \dots \bar{A}_n$ where $W \setminus \{\beta_0\} \subseteq \bar{F}_0 \subseteq Y_0$. If $W \subseteq \bar{F}_0$, we are done. Otherwise, proceeding like the base case, we construct a step sequence v_3 such that $\bar{v}_3 = \bar{F}_0 \bar{F}_1 \bar{v}_3 \equiv \bar{A}_1 \dots \bar{A}_n$ and $\{\beta_0\} \subseteq \bar{F}_1 \subseteq Y_0$. Since $\bar{F}_0 \subseteq Y_0$, we have $W \subseteq \bar{F}_0 \cup \bar{F}_1 \subseteq Y_0$. Then similarly to how we proved (11.11), we can show that $\forall \alpha \in \bar{F}_0. \forall \beta \in \bar{F}_1. \{\alpha\}\{\beta\} \equiv \{\alpha, \beta\}$. This means that for all $\alpha \in \bar{F}_0$ and $\beta \in \bar{F}_1$, $(l(\alpha), l(\beta)) \in \text{ser}$. Hence, $\bar{F}_0 \bar{F}_1 \equiv \bar{F}_0 \cup \bar{F}_1$. Hence, there is a step sequence v_4 such that $\bar{v}_4 = (\bar{F}_0 \cup \bar{F}_1) \bar{v}_4 \equiv \bar{A}_1 \dots \bar{A}_n$ and $W \subseteq (\bar{F}_0 \cup \bar{F}_1) \subseteq Y_0$.

Thus, we have shown (11.9). So by choosing $W = Y_0$, we get a step sequence v such that $\bar{v} = W_0 \bar{v}_0 \equiv \bar{A}_1 \dots \bar{A}_n$ and $Y_0 \subseteq W_0 \subseteq Y_0$. Hence, $\bar{v} = W_0 \bar{v}_0 \equiv \bar{A}_1 \dots \bar{A}_n$. Thus, $v = B_0 v_0 \equiv A_1 \dots A_n$. But since $B_0 <^{st} A_1$, this contradicts the fact that $A_1 \dots A_n$ is the least element of $[s]$ w.r.t. $<^{lex}$. Hence, A_1 is the least element of $\{l[Y] \mid Y \in Z(\Sigma)\}$ w.r.t. $<^{st}$.

We now prove that A_i is the least element of $\{l[Y] \mid Y \in Z(\Sigma \setminus \biguplus(\bar{A}_1 \dots \bar{A}_{i-1}))\}$ w.r.t. $<^{st}$ by using induction on n , the number of steps of $A_1 \dots A_n$. If $n = 0$, we are done. If $n > 0$, then we have just shown that A_1 is the least element of $\{l[Y] \mid Y \in Z(\Sigma)\}$ w.r.t. $<^{st}$. By applying the induction hypothesis on $p = \bar{A}_2 \dots \bar{A}_n$, $\Sigma_p = \Sigma \setminus \bar{A}_1$, and its gso-structure $(\Sigma_p, \diamond \cap (\Sigma_p \times \Sigma_p), \sqsubset \cap (\Sigma_p \times \Sigma_p))$, we get A_i is the least element of the set $\{l[Y] \mid Y \in Z(\Sigma \setminus \biguplus(\bar{A}_1 \dots \bar{A}_{i-1}))\}$ w.r.t. $<^{st}$ for all $i \geq 2$. \square

Theorem 11.3. *Let s and t be step sequences over a g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$. Then $s \equiv t$ iff $G^{\{s\}} = G^{\{t\}}$.*

PROOF. (\Rightarrow) If $s \equiv t$, then $[s] = [t]$. Hence, by Theorem 11.2, $G^{\{s\}} = G^{\{t\}}$.

(\Leftarrow) By Lemma 11.1, we can use $G^{\{s\}}$ to construct a unique element w_1 such that w_1 is the least element of $[s]$ w.r.t. $<^{lex}$, and then use $G^{\{t\}}$ to construct a unique element w_2 that is the least element of $[t]$ w.r.t. $<^{lex}$. But since $G^{\{s\}} = G^{\{t\}}$ and the construction is unique, we get $w_1 = w_2$. Hence, $s \equiv t$. \square

Theorem 11.3 justifies the following definition:

Definition 11.6. For every g-comtrace $[s]$, $G_{[s]} = G^{\{s\}} = (\Sigma_s, \prec_s \cup \diamond_s, \prec_s \cup \sqsubset_s)^{\bowtie}$ is the gso-structure induced by the g-comtrace $[s]$. \blacksquare

To end this section, we prove two major results. Theorem 11.4 says that the stratified extensions of the gso-structure induced by a g-comtrace $[t]$ are exactly those generated by the step sequences in $[t]$. Theorem 11.5 says that the gso-structure induced by a g-comtrace is uniquely identified by any of its stratified extensions.

Lemma 11.2. *Let $s, t \in \mathbb{S}^*$ and $\triangleleft_s \in \text{ext}(G^{\{t\}})$. Then $G^{\{s\}} = G^{\{t\}}$.* \square

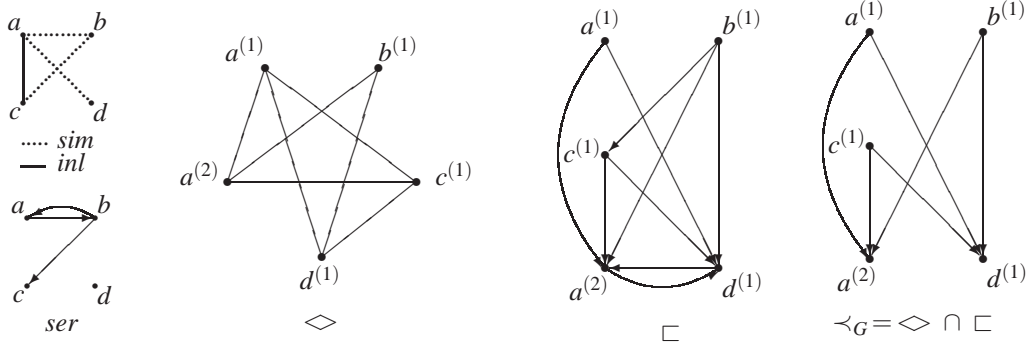


Figure 4: A g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$, where $E = \{a, b, c, d\}$, the gso-structure $G = (X, \diamond, \square)$ and the relation $\prec_G = \diamond \cap \square$ defined by the g-comtrace $[\{a, b\}\{c\}\{a, d\}] = \{\{a, b\}\{c\}\{a, d\}, \{a\}\{b\}\{c\}\{a, d\}, \{a\}\{b, c\}\{a, d\}, \{b\}\{a\}\{c\}\{a, d\}, \{b\}\{c\}\{a\}\{a, d\}, \{b, c\}\{a\}\{a, d\}\}$.

The proof of the above lemma is quite long and based heavily on Definition 11.3. It requires a separate analysis of many cases and was moved to Appendix B.

Theorem 11.4. *Let $s, t \in \mathbb{S}^*$. Then $\text{ext}(G^{\{s\}}) = \{\triangleleft_u \mid u \in [s]\}$.*

PROOF. (\subseteq) Suppose $\triangleleft \in \text{ext}(G^{\{s\}})$. By Proposition 11.6, there is a step sequence u such that $\triangleleft_u = \triangleleft$. Hence, by Lemma 11.2, we have $G^{\{u\}} = G^{\{s\}}$, which by Theorem 11.3 implies that $u \equiv s$. Hence, $\text{ext}(G^{\{s\}}) \supseteq \{\triangleleft_u \mid u \in [s]\}$.

(\supseteq) If $u \in [s]$, then it follows from Theorem 11.3 that $G^{\{u\}} = G^{\{s\}}$. This and Proposition 11.5 imply $\triangleleft_u \in \text{ext}(G^{\{s\}})$. Hence, $\text{ext}(G^{\{s\}}) \supseteq \{\triangleleft_u \mid u \in [s]\}$. \square

Theorem 11.5. *Let $s, t \in \mathbb{S}^*$ and $\text{ext}(G^{\{s\}}) \cap \text{ext}(G^{\{t\}}) \neq \emptyset$. Then $s \equiv t$.*

PROOF. Let $\triangleleft \in \text{ext}(G^{\{s\}}) \cap \text{ext}(G^{\{t\}})$. By Proposition 11.6, there is a step sequence u such that $\triangleleft_u = \triangleleft$. By Lemma 11.2, we have $G^{\{s\}} = G^{\{u\}} = G^{\{t\}}$. This and Theorem 11.3 yields $s \equiv t$. \square

Summing up, we have proved the equivalence of Theorem 10.2 but for g-comtraces. In fact, Theorem 10.2 is a straightforward consequence of this section for $\text{inl} = \emptyset$.

12. Generalized Comtraces Representing Finite Generalized Stratified Order Structures

In this section we will show that every finite gso-structure can be represented by a g-comtrace. Let $G = (X, \diamond, \square)$ be a finite gso-structure and let $\Delta = \{\Omega_{\triangleleft} \mid \triangleleft \in \text{ext}(G)\}$ be the set of all step sequences defined by the elements of $\text{ext}(G)$. Note that each “event” in X only occurs in each sequence of Δ at most once. Hence, we do not need to enumerate the sequences in Δ or to use the label function l in the definitions and proofs.

We will start with the definition of relations *simultaneity*, *serializability* and *interleaving* induced by a given gso-structure $G = (X, \diamond, \square)$. The following definition is an extension of Definition 10.4 to gso-structures.

Definition 12.1. For each $a, b \in X$:

1. $(a, b) \in \text{sim}_G \stackrel{\text{df}}{\iff} \neg(a \diamond b),$
2. $(a, b) \in \text{ser}_G \stackrel{\text{df}}{\iff} \neg(a \diamond b) \wedge \neg(b \sqsubset a),$
3. $(a, b) \in \text{inl}_G \stackrel{\text{df}}{\iff} a \diamond b \wedge \neg(a \sqsubset b \vee b \sqsubset a).$ ■

Definition 12.1 is a generalization of Definition 10.4 as shown in the following proposition. We recall that from Definition 4.3 we defined $\prec_G = \diamond \cap \sqsubset$.

Proposition 12.1.

1. $(a, b) \in \text{sim}_G \iff a \frown_{\prec_G} b,$
2. $(a, b) \in \text{ser}_G \iff a \frown_{\prec_G} b \wedge \neg(b \sqsubset a),$
3. $\text{inl}_G = \emptyset.$ □

The names *simultaneity*, *serializability* and *interleaving* are justified by the following result, which shows the connection between these three relations and the stratified extensions of the gso-structure G .

Proposition 12.2. For each $a, b \in X$, we have

1. $(a, b) \in \text{sim}_G \iff \exists \triangleleft \in \text{ext}(G). a \frown_{\triangleleft} b,$
2. $(a, b) \in \text{ser}_G \iff (a, b) \in \text{sim} \wedge (\exists \triangleleft \in \text{ext}(G). a \triangleleft b),$
3. $(a, b) \in \text{inl}_G \iff (a, b) \notin \text{sim} \wedge (\exists \triangleleft \in \text{ext}(G). a \triangleleft b) \wedge (\exists \triangleleft \in \text{ext}(G). b \triangleleft a).$
4. $(\exists \triangleleft \in \text{ext}(G). a \triangleleft b) \wedge (a, b) \notin \text{ser}_G \iff (\forall \triangleleft \in \text{ext}(G). a \triangleleft^{\text{sym}} b) \iff a \diamond b$

PROOF. (1), (2) and (3) follows from Theorem 4.3; (4) follows from (1), (2) and Theorem 4.3. □

Definition 12.2. For each stratified order $\triangleleft \in \text{ext}(G)$, we define

$$G^{\{\triangleleft\}} \stackrel{\text{df}}{=} \left(X, (\triangleleft \setminus \text{ser}_G)^{\text{sym}} \cup \text{inl}_G, \triangleleft \cap (\text{ser}_G^{-1} \cup \text{inl}_G) \right) \quad \blacksquare$$

The following proposition shows that $G^{\{\triangleleft\}}$ and G are identical gso-structures. Similarly to the relationship between $S^{\{\triangleleft\}}$ and S , it is also interesting to observe that we do not need commutative closure to build G from a stratified order $\triangleleft \in \text{ext}(G)$. The reason behind this simplification is explained in the following proposition, which is a generalization of Proposition 10.2.

Proposition 12.3. For every $\triangleleft \in \text{ext}(G)$, we have:

1. $(\triangleleft \setminus \text{ser}_G)^{\text{sym}} \cup \text{inl}_G = \diamond = \diamond \setminus \text{ser}_G^{\text{sym}},$
2. $\triangleleft \cap (\text{ser}_G^{-1} \cup \text{inl}_G) = \sqsubset = \sqsubset \setminus (\text{ser}_G^{-1} \cup \text{inl}_G),$
3. $G^{\{\triangleleft\}} = (X, \diamond, \sqsubset),$
4. $\prec_G = \triangleleft \setminus (\text{ser}_G \cup \text{inl}_G),$
5. $\diamond = \prec_G^{\text{sym}} \cup \text{inl}_G.$

PROOF. 1. For every $a, b \in X$, we have

$$\begin{aligned}
& (a, b) \in (\triangleleft \setminus \text{ser}_G)^{\text{sym}} \cup \text{inl}_G \\
& \iff (a \triangleleft b \wedge (a, b) \notin \text{ser}_G) \vee (b \triangleleft a \wedge (b, a) \notin \text{ser}_G) \vee (a, b) \in \text{inl}_G \\
& \iff a \diamond b \vee (a, b) \in \text{inl}_G \quad \langle \text{Proposition 12.2(4)} \rangle \\
& \iff a \diamond b \vee (a \diamond b \wedge \neg(b \sqsubseteq a \vee a \sqsubseteq b)) \quad \langle \text{Definition 12.1(3)} \rangle \\
& \iff a \diamond b
\end{aligned}$$

Hence, the first equality holds. The second equality immediately follows since $\text{inl}_G \cap \text{ser}_G = \emptyset$.

2. For every $a, b \in X$, we have

$$\begin{aligned}
& (a, b) \in \triangleleft^\cap \setminus (\text{ser}_G^{-1} \cup \text{inl}_G) \\
& \iff a \triangleleft^\cap b \wedge (a \diamond b \vee a \sqsubseteq b) \wedge (\neg(a \diamond b) \vee a \sqsubseteq b \vee b \sqsubseteq a) \quad \langle \text{Definition 12.1} \rangle \\
& \iff a \triangleleft^\cap b \wedge (a \sqsubseteq b \vee (a \diamond b \wedge b \sqsubseteq a)) \\
& \iff (a \triangleleft^\cap b \wedge a \sqsubseteq b) \vee (a \triangleleft^\cap b \wedge b \prec_G a) \\
& \iff a \sqsubseteq b \vee \text{False} \quad \langle \text{Theorem 4.3} \rangle
\end{aligned}$$

3. Follows from (1) and (2).

4. For every $a, b \in X$, we have

$$\begin{aligned}
& (a, b) \in \triangleleft \setminus (\text{ser}_G \cup \text{inl}_G) \\
& \iff a \triangleleft b \wedge (a \sqsubseteq b \vee (a \diamond b \wedge b \sqsubseteq a)) \quad \langle \text{Similarly to proof of (2)} \rangle \\
& \iff a \triangleleft b \wedge (a \prec_G b \vee a \sqsubseteq^{\text{sym}} b) \\
& \iff (a \triangleleft b \wedge a \prec_G b) \vee (a \triangleleft b \wedge a \sqsubseteq^{\text{sym}} b) \\
& \iff a \prec_G b \vee \text{False} \quad \langle \text{Theorem 4.3} \rangle
\end{aligned}$$

5. Follows from (1), (4) and the fact that inl_G is symmetric. \square

We have just shown that every so-structure $G = (X, \diamond, \sqsubseteq)$ is equal to $G^{\{\triangleleft\}}$ for any stratified order $\triangleleft \in \text{ext}(G)$. Note that since the proof does not assume that G is finite, Proposition 12.3 also holds when G is an *infinite* gso-structure.

Before stating the main theorem of this section, we need the following definition.

Definition 12.3. For each finite gso-structure $G = (X, \diamond, \sqsubseteq)$, we define:

1. $\Theta_G \stackrel{\text{df}}{=} (X, \text{sim}_G, \text{ser}_G, \text{inl}_G)$,
2. $\text{g}\mathfrak{C}(G) \stackrel{\text{df}}{=} \{\Omega_\triangleleft \mid \triangleleft \in \text{ext}(G)\}$. \blacksquare

Observe that $\text{ser}_G \subseteq \text{sim}_G$, the relations sim and inl are symmetric, $\text{sim}_G \cap \text{inl}_G = \emptyset$, and all three relations are irreflexive, so Θ_G is a *g-comtrace alphabet*. Hence we can define the relations $\approx_{\{\text{ser}, \text{inl}\}}$ and $\equiv_{\{\text{ser}, \text{inl}\}}$ with respect to the g-comtrace alphabet Θ_G . We will call $\text{g}\mathfrak{C}(G)$ the *g-comtrace generated by the gso-structure G* . Theorem 12.1 below will justify this name.

Lemma 12.1. Let $G = (X, \diamond, \sqsubseteq)$ be a finite gso-structure and let $\Delta = \{\Omega_\triangleleft \mid \triangleleft \in \text{ext}(G)\}$. Then we have:

1. Δ is a set of step sequences over Θ_G .
2. If $u \in \Delta$, then

$$\diamondsuit_u = \text{inl}_G \quad (12.1)$$

$$\sqsubset_u = \triangleleft^\cap \setminus (\text{ser}_G^{-1} \cup \text{inl}_G) = \sqsubset \quad (12.2)$$

$$\prec_u = \triangleleft^\cap \setminus (\text{ser}_G \cup \text{inl}_G) = \prec_G \quad (12.3)$$

PROOF (PROOF OF LEMMA 12.1).

1. We need to check that every element of Δ is a step sequence over the g-contrace alphabet Θ_G . Let $\Omega_\triangleleft = A_1 \dots A_n \in \Delta$. Then since $\triangleleft \in \text{ext}(G)$, for each A_i ($1 \leq i \leq n$), we have if $a, b \in A_i$ and $a \neq b$, then $a \triangleleft b$. Hence, from Proposition 12.2(1), we have $(a, b) \in \text{sim}_G$.
2. The equality from (12.1) immediately follows from (11.1) of Definition 11.3.

The first and second equalities from (12.2) follow from (11.2) of Definition 11.3 and Proposition 12.3(2).

The second equality (12.3) follows from Proposition 12.3(3). It remains to show the first equality of (12.3).

(\supseteq) Follows from (11.2) of Definition 11.3.

(\subseteq) We let \triangleleft be the stratified order in $\text{ext}(G)$ such that $u = \Omega_\triangleleft$. It suffices to show that for every $a, b \in X$, the fact that $a \triangleleft b$ and

$$\begin{aligned} & (a, b) \in \diamondsuit_u \cap ((\sqsubset_u^*)^\cap \circ \diamondsuit_u^C \circ (\sqsubset_u^*)^\cap) \\ \vee & \left((a, b) \in \text{ser}_G \wedge \exists c, d \in X. \left(\begin{array}{l} c \triangleleft_u d \wedge (c, d) \notin \text{ser}_G \\ \wedge \quad a \sqsubset_u^* c \sqsubset_u^* b \wedge a \sqsubset_u^* d \sqsubset_u^* b \end{array} \right) \right) \end{aligned}$$

leads to a contradiction. There are two cases to consider:

- (a) If $a \triangleleft b$ and $(a, b) \in \diamondsuit_u \cap ((\sqsubset_u^*)^\cap \circ \diamondsuit_u^C \circ (\sqsubset_u^*)^\cap)$, then there must be $c, d \in X$ such that $a \sqsubset_u^* c$ and $d \sqsubset_u^* b$ and $(c, d) \notin \text{inl}_G$. Since we know from (2) that $\sqsubset_u = \sqsubset$, we have $a \sqsubset^* c$ and $d \sqsubset^* b$. Since $\sqsubset^* = \sqsubset \cup \text{id}_X$, there are three cases to consider. If $a \sqsubset c \wedge c \sqsubset a \wedge d \sqsubset b \wedge b \sqsubset d$, then it follows that $\forall \triangleleft \in \text{ext}(G). (a \triangleleft c \wedge d \triangleleft b)$. Since $a \diamondsuit_u b$ implies that $(a, b) \in \text{inl}_G$, which means $(a, b) \notin \text{sim}_G \wedge (\exists \triangleleft \in \text{ext}(S). a \triangleleft b) \wedge (\exists \triangleleft \in \text{ext}(S). b \triangleleft a)$. Thus, $(c, d) \notin \text{sim}_G \wedge (\exists \triangleleft \in \text{ext}(S). c \triangleleft d) \wedge (\exists \triangleleft \in \text{ext}(S). d \triangleleft c)$, contradicting that $(c, d) \notin \text{inl}_G$. Similarly, we can show that the remaining two cases ($a = c \wedge d \sqsubset b \wedge b \sqsubset d$) and ($a \sqsubset c \wedge c \sqsubset a \wedge d = b$) also lead to a contradiction.
- (b) If $a \triangleleft_u b$ and $(a, b) \in \text{ser}_G$ and $\exists c, d \in X. \left(\begin{array}{l} c \triangleleft_u d \wedge (c, d) \notin \text{ser}_G \\ \wedge \quad a \sqsubset_u^* c \sqsubset_u^* b \wedge a \sqsubset_u^* d \sqsubset_u^* b \end{array} \right)$, then since we know from (2) that $\sqsubset_u = \sqsubset$, we have $a \sqsubset^* c \sqsubset^* b$ and $a \sqsubset^* d \sqsubset^* b$. Since we have $\sqsubset^* = \sqsubset \cup \text{id}_X$, it follows that $\forall w \in \Delta. (\text{pos}_w(a) \leq \text{pos}_w(c) \leq \text{pos}_w(b) \wedge \text{pos}_w(a) \leq \text{pos}_w(d) \leq \text{pos}_w(b))$. But since $(a, b) \in \text{ser}_G$, there is some $v \in \Delta$ such that $\text{pos}_v(a) = \text{pos}_v(b)$. So $\text{pos}_v(c) = \text{pos}_v(d)$, i.e., $(c, d) \in \text{sim}_G$. This and $c \triangleleft_u d$ imply that $(c, d) \in \text{ser}_G$, a contradiction. \square

Theorem 12.1. Let $G = (X, \diamondsuit, \sqsubset)$ be a finite gso-structure. For every $\triangleleft \in \text{ext}(G)$, we have:

1. $G_{[\Omega_\triangleleft]} = G^{\{\Omega_\triangleleft\}} = G^{\{\triangleleft\}} = G$,
2. $\text{g}\mathcal{C}(G) = [\Omega_\triangleleft]$.

\square

PROOF. 1. Given a stratified order $\triangleleft \in G$. Let $u = \Omega_{\triangleleft}$. Since $u = \Omega_{\triangleleft}$ is a step sequence over Θ_G , by Definition 11.3, we can construct $G^{\{u\}} = (X, \prec_u \cup \diamond_u, \prec_u \cup \sqsubset_u)^\infty$. Thus, by Lemma 12.1, we get $G^{\{u\}} = (X, \prec_G \cup \text{inl}_G, \prec_G \cup \sqsubset)^\infty = (X, \prec_G \cup \text{inl}_G, \sqsubset)^\infty$. Using \bowtie -closure from Definition 11.1, we get $R_3 = (\prec_G \cup \text{inl}_G) \cap \sqsubset^* = (\prec_G \cup \text{inl}_G) \cap \sqsubset$. But since $\text{inl}_G \subseteq \diamond$ and $\prec_G = \diamond \cap \sqsubset$, it follows that $R_3 = \prec_G$. We then apply the \diamond -closure to get $(X, \prec_{R_3 \sqsubset}, \sqsubset_{R_3 \sqsubset}) = (X, \prec_G, \sqsubset_G)^\diamond$. Since (X, \prec_G, \sqsubset) is a so-structure, it follows from Theorem 10.1 that $\prec_{R_3 \sqsubset} = \prec_G$ and $\sqsubset_{R_3 \sqsubset} = \sqsubset$. Thus, we have:

$$\begin{aligned}
 G_{[u]} &= G^{\{u\}} && \langle \text{Theorem 11.2} \rangle \\
 &= (X, \prec_G \cup \text{inl}_G, \prec_G \cup \sqsubset)^\infty = (X, \prec_G^{\text{sym}} \cup \text{inl}_G, \sqsubset) \\
 &= (X, \diamond, \sqsubset) = G^{\{\triangleleft\}} = G && \langle \text{Proposition 12.3(3, 4)} \rangle
 \end{aligned}$$

2. From (1) and Theorem 11.4, we know that $\text{ext}(G_{[\Omega_{\triangleleft}]}) = \text{ext}(G) = \{\triangleleft_u \mid u \in [\Omega_{\triangleleft}]\}$. But this implies that $[\Omega_{\triangleleft}] = \{\Omega_{\blacktriangleleft} \mid \blacktriangleleft \in \text{ext}(G)\} = \mathfrak{gC}(G)$. \square

Together with Theorem 11.4, Theorem 12.1 ensures that g-comtraces and finite so-structure are in fact equivalent models.

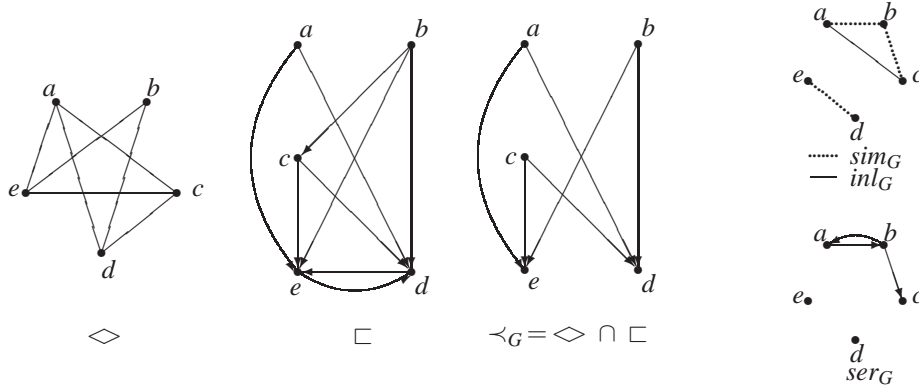


Figure 5: A gso-structure $G = (X, \diamond, \sqsubset)$, where $X = \{a, b, c, d, e\}$ and the relations \prec_G , sim_G , ser_G and inl_G . The gso-structure G defines the g-comtrace $[\{a, b\}\{c\}\{e, d\}] = \{\{a, b\}\{c\}\{e, d\}, \{a\}\{b\}\{c\}\{e, d\}, \{a\}\{b, c\}\{e, d\}, \{b\}\{a\}\{c\}\{e, d\}, \{b\}\{c\}\{a\}\{e, d\}, \{b, c\}\{a\}\{e, d\}\}$.

13. Conclusion and Future Work

The concept of a comtrace is revisited and its extension, the g-comtrace, is introduced. Both comtraces and g-comtraces can be seen as generalizations of Mazurkiewicz traces. We analyzed some algebraic and linguistic properties of comtraces and g-comtraces, where an interesting application of the algebraic properties of comtraces is the proof of the uniqueness of comtrace canonical representation. We study the canonical representations of traces, comtraces and g-comtraces and their mutual relationships in a more unified framework. We observe that traces and

comtraces have a natural unique canonical form which corresponds to their maximal concurrent representation, while the only unique canonical representation of a g-comtrace is by choosing the lexicographically least element of the g-comtrace. We also revisit the mutual relationship between comtraces and so-structures from [12] and show that each comtrace can be uniquely represented by a so-structure.

The most important contribution of this paper is the study of the mutual relationship between g-comtraces and gso-structures. The major technical results, Theorems 11.4 and 12.1, can be seen as the generalizations of Szpilrajn's Theorem in the context of g-comtraces. Furthermore, Theorems 11.4 and 12.1 ensure that g-comtraces and finite gso-structures can uniquely represent one another. We believe the reason the proofs of 11.4 and 12.1 are more technical than similar theorems of comtraces is that both comtraces and so-structures satisfy paradigm π_3 while g-comtraces and gso-structures do not. Intuitively, what paradigm π_3 really says is that the underlying structure is really a partial order. For comtraces and so-structures, we did augment some more priority relationships into the incomparable elements with respect to the standard causal partial order to produce the not later than relation. Note that this process might introduce cycles into the graph of the "not later than" relation. However, it is important to observe that any two distinct elements lying on a cycle of the "not later than" relations must belong to a synchronous step. Thus, if we collapse each synchronous set into a single vertex, then the resulting relation is a partial order. When paradigm π_3 is not satisfied, we have much more than a partial order structure, and hence common techniques that depend too much on the underlying partial order structure of comtraces and so-structures will not work for g-comtraces and gso-structures.

Despite some obvious advantages, for instance very handy composition and no need to use labels, quotient monoids (perhaps with some exception of traces) are much less popular for analyzing issues of concurrency than their relational counterparts such as partial orders, so-structures, occurrence graphs, etc. We believe that in many cases, advanced quotient monoids, e.g., comtraces and g-comtraces, could provide simpler and more adequate models of concurrent histories than their relational equivalences.

Much harder future tasks are in the area of comtrace and g-comtrace languages where major problems as recognisability [25], acceptability [30], etc. are still open.

References

- [1] W. Bartussek, D. L. Parnas, Using Assertions About Traces to Write Abstract Specifications for Software Modules, *Lecture Notes in Computer Science* 65, Springer 1978, pp. 211–236.
- [2] P. Cartier and D. Foata, *Problèmes combinatoires de commutation et ré-arrangements*, *Lecture Notes in Mathematics* 85, Springer 1969.
- [3] P. M. Cohn, *Universal Algebra*, D. Reidel 1981.
- [4] R. Davalliers, R. Janicki, M. Koutny, P. E. Lauer, Concurrent and Maximally Concurrent Evolution of Non-sequential Systems, *Theoretical Computer Science* 43:213-238, 1986.
- [5] V. Diekert and G. Rozenberg (eds.), *The Book of Traces*. World Scientific 1995.
- [6] H. Gaifman and V. Pratt, Partial Order Models of Concurrency and the Computation of Function, *Proc. of LICS'87*, pp. 72-85.
- [7] G. Guo and R. Janicki, Modelling Concurrent Behaviours by Commutativity and Weak Causality Relations, *Proc. of AMAST'02, LNCS 2422* (2002), 178-191.
- [8] C. A. R. Hoare, *Communicating Sequential Processes*, Prentice-Hall, 1986.
- [9] R. Janicki. Relational Structures Model of Concurrency. *Acta Informatica*, 45(4): 279-320, 2008.
- [10] R. Janicki and M. Koutny, Invariants and Paradigms of Concurrency Theory, *LNCS 506*, Springer 1991, pp. 59-74.
- [11] R. Janicki and M. Koutny, Structure of Concurrency, *Theoretical Computer Science*, 112(1): 5–52, 1993.
- [12] R. Janicki and M. Koutny, Semantics of Inhibitor Nets, *Information and Computation*, 123(1): 1–16, 1995.
- [13] R. Janicki and M. Koutny, Fundamentals of Modelling Concurrency Using Discrete Relational Structures, *Acta Informatica*, 34: 367–388, 1997.

- [14] R. Janicki and M. Koutny, On Causality Semantics of Nets with Priorities, *Fundamenta Informaticae* 34: 222–255, 1999.
- [15] R. Janicki, P. E. Lauer, *Specification and Analysis of Concurrent Systems: The COSY Approach*, Springer 1992.
- [16] R. Janicki and D. T. M. Lê, Modelling Concurrency with Quotient Monoids, Proc of PETRI NETS 2008, *LNCS* 5062, Springer 2008, pp. 251–269.
- [17] G. Juhás, R. Lorenz, S. Mauser, Causal Semantics of Algebraic Petri Nets distinguishing Concurrency and Synchronicity, *Fundamenta Informatica* 86(3): 255–298, 2008.
- [18] G. Juhás, R. Lorenz, S. Mauser, Synchronous + Concurrent + Sequential = Earlier Than + Not Later Than, *Proc. of ACSD'06* (Application of Concurrency to System Design), Turku, Finland 2006, pp. 261–272, IEEE Press.
- [19] G. Juhás, R. Lorenz, C. Neumair, Synthesis of Controlled Behaviour with Modules of Signal Nets, *LNCS* 3099, Springer 2004, pp. 233–257.
- [20] H. C. M. Kleijn and M. Koutny, Process Semantics of General Inhibitor Nets, *Information and Computation*, 190:18–69, 2004.
- [21] J. Kleijn and M. Koutny, Formal Languages and Concurrent Behaviour, *Studies in Computational Intelligence*, 113:125–182, 2008.
- [22] D. T. M. Lê., Studies in Comtrace Monoids, Master Thesis, Dept. of Computing and Software, McMaster University, Hamilton, Ontario, Canada, August 2008.
- [23] A. Mazurkiewicz, Concurrent Program Schemes and Their Interpretation, TR DAIMI PB-78, Comp. Science Depart., Aarhus University, 1977.
- [24] A. Mazurkiewicz, Introduction to Trace Theory, in [5], pp. 3–42.
- [25] E. Ochmański, Recognizable Trace Languages, in [5], pp. 167–204.
- [26] M. Pietkiewicz-Koutny, The Synthesis Problem for Elementary Net Systems, *Fundamenta Informaticae* 40(2,3):310–327, 1999.
- [27] M. W. Shields, Adequate Path Expressions, *LNCS* 70, Springer 1979, pp. 249–265.
- [28] E. Szpilrajn, Sur l'extension de l'ordre partiel, *Fundamenta Mathematicae* 16 (1930), 386–389.
- [29] W. Vogler, A Generalization of Trace Theory, *RAIRO Inform. Théor. Appl.* 25(2):147–156, 1991.
- [30] W. Zielonka, Notes on Finite Asynchronous Automata, *RAIRO Inform. Théor. Appl.* 21:99–135, 1987.

Appendix A: Proof of Proposition 11.4

Proposition A.1. *Let u be a step sequence over a g -comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$ and $\alpha, \beta \in \Sigma_u$ such that $l(\alpha) = l(\beta)$. Then*

1. $\text{pos}_u(\alpha) \neq \text{pos}_u(\beta)$
2. *If $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$ and v is a step sequence satisfying $v \equiv u$, then $\text{pos}_v(\alpha) < \text{pos}_v(\beta)$.*

PROOF. 1. Follows from the fact that *sim* is irreflexive.

2. It suffices to show that if $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$ and $\bar{v} \approx \bar{u}$, then $\text{pos}_v(\alpha) < \text{pos}_v(\beta)$. But this is clear from Proposition 6.1 and the fact that *ser* and *inl* are irreflexive. \square

Proposition A.2. *Let u, w be step sequences over a g -comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$ such that $u(\approx \cup \approx^{-1})w$. Then*

1. *If $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$ and $\text{pos}_w(\alpha) > \text{pos}_w(\beta)$ then there are x, y, A, B such that $\bar{u} = \bar{x}\bar{A}\bar{B}\bar{y}(\approx \cup \approx^{-1})\bar{x}\bar{B}\bar{A}\bar{y} = \bar{w}$ and $\alpha \in \bar{A}, \beta \in \bar{B}$.*
2. *If $\text{pos}_u(\alpha) = \text{pos}_u(\beta)$ and $\text{pos}_w(\alpha) > \text{pos}_w(\beta)$ then there are x, y, A, B, C such that $\bar{u} = \bar{x}\bar{A}\bar{y} \approx \bar{x}\bar{B}\bar{C}\bar{y} = \bar{w}$ and $\beta \in \bar{B}$ and $\alpha \in \bar{C}$.*

PROOF. 1. Assume that $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$ and $\text{pos}_w(\alpha) > \text{pos}_w(\beta)$. Since $u(\approx \cup \approx^{-1})w$, we observe that

- If $\bar{u} = \bar{s}\bar{D}\bar{t} \approx \bar{s}\bar{E}\bar{F}\bar{t} = \bar{w}$, then $\forall \alpha, \beta \in \text{supp}(\bar{u}), \text{pos}_u(\alpha) < \text{pos}_u(\beta) \implies \text{pos}_w(\alpha) < \text{pos}_w(\beta)$.

- If $\bar{u} = \bar{s}\bar{D}\bar{E}\bar{t} \approx \bar{s}\bar{F}\bar{t} = \bar{w}$, then $\forall \alpha, \beta \in \uplus(\bar{u}), pos_u(\alpha) < pos_u(\beta) \implies pos_w(\alpha) \leq pos_w(\beta)$.

Either case contradicts the assumption that $pos_w(\alpha) > pos_w(\beta)$. Hence, it must be the case that $\bar{u} = \bar{x}\bar{A}\bar{B}\bar{y}(\approx \cup \approx^{-1})\bar{x}\bar{B}\bar{A}\bar{y} = \bar{w}$ for some x, y, A, B . We will show that $\alpha \in \bar{A}$ and $\beta \in \bar{B}$. Suppose that $\alpha \notin \bar{A}$ or $\beta \notin \bar{B}$. Then

- If $\alpha \notin \bar{A}$, then $\forall \alpha, \beta \in \uplus(\bar{x}) \cup \bar{B} \cup \uplus(\bar{y}), pos_u(\alpha) < pos_u(\beta) \implies pos_w(\alpha) < pos_w(\beta)$.
- If $\beta \notin \bar{B}$, then $\forall \alpha, \beta \in \uplus(\bar{x}) \cup \bar{A} \cup \uplus(\bar{y}), pos_u(\alpha) < pos_u(\beta) \implies pos_w(\alpha) < pos_w(\beta)$.

Either case contradicts that $pos_w(\alpha) > pos_w(\beta)$. Hence, we have $\bar{u} = \bar{x}\bar{A}\bar{B}\bar{y}(\approx \cup \approx^{-1})\bar{x}\bar{B}\bar{A}\bar{y} = \bar{w}$, where $\alpha \in \bar{A}$ and $\beta \in \bar{B}$ as desired.

2. Can be shown in a similar way to (1). \square

Proposition A.3. *Let s be a step sequence over a g -comtrace alphabet (E, sim, ser, inl) . If $\alpha, \beta \in \Sigma_s$, then*

1. $\alpha \diamond_s \beta \implies \forall u \in [s]. pos_u(\alpha) \neq pos_u(\beta)$,
2. $\alpha \sqsubset_s \beta \implies \forall u \in [s]. pos_u(\alpha) \leq pos_u(\beta)$,
3. $\alpha \prec_s \beta \implies \forall u \in [s]. pos_u(\alpha) < pos_u(\beta)$.

PROOF. 1. Assume that $\alpha \diamond_s \beta$. Then, by (11.1), $(l(\alpha), l(\beta)) \in inl$. This implies that $l(\alpha) \neq l(\beta)$, so $\alpha \neq \beta$. Also since $inl \cap sim = \emptyset$, there is no step A where $\{l(\alpha), l(\beta)\} \in A$. Hence, $\forall u \in [s]. pos_u(\alpha) \neq pos_u(\beta)$.

2. Assume that $\alpha \sqsubset_s \beta$. Suppose that $\exists u \in [s]. pos_u(\alpha) > pos_u(\beta)$. Then there must be some $u_1, u_1 \in [s]$ such that $u_1(\approx \cup \approx^{-1})u_2$ and $pos_{u_1}(\alpha) \leq pos_{u_1}(\beta)$ and $pos_{u_2}(\alpha) > pos_{u_2}(\beta)$. There are two cases:

- If $pos_{u_1}(\alpha) < pos_{u_1}(\beta)$ and $pos_{u_2}(\alpha) > pos_{u_2}(\beta)$, then by Proposition A.2(1) there are x, y, A, B such that $\bar{u}_1 = \bar{x}\bar{A}\bar{B}\bar{y}(\approx \cup \approx^{-1})\bar{x}\bar{B}\bar{A}\bar{y} = \bar{u}_2$ and $\alpha \in \bar{A}, \beta \in \bar{B}$. Hence, $(l(\alpha), l(\beta)) \in inl$, which by (11.2) contradicts that $\alpha \sqsubset_s \beta$.
- If $pos_{u_1}(\alpha) = pos_{u_1}(\beta)$ and $pos_{u_2}(\alpha) > pos_{u_2}(\beta)$, then it follows from Proposition A.2(2) that there are x, y, A, B, C such that $\bar{u}_1 = \bar{x}\bar{A}\bar{y} \approx \bar{x}\bar{B}\bar{C}\bar{y} = \bar{u}_2$ and $\beta \in \bar{B}$ and $\alpha \in \bar{C}$. Thus, $(l(\beta), l(\alpha)) \in ser$, which by (11.2) contradicts that $\alpha \sqsubset_s \beta$.

3. Assume that $\alpha \prec_s \beta$. Suppose that $\exists u \in [s]. pos_u(\alpha) \geq pos_u(\beta)$. Then must be some $u_1, u_1 \in [s]$ such that $u_1(\approx \cup \approx^{-1})u_2$ and $pos_{u_1}(\alpha) < pos_{u_1}(\beta)$ and $pos_{u_2}(\alpha) \geq pos_{u_2}(\beta)$. There are two cases:

- If $pos_{u_1}(\alpha) < pos_{u_1}(\beta)$ and $pos_{u_2}(\alpha) = pos_{u_2}(\beta)$, then it follows from Proposition A.2(2) that there are x, y, A, B, C such that $\bar{u}_2 = \bar{x}\bar{A}\bar{y} \approx \bar{x}\bar{B}\bar{C}\bar{y} = \bar{u}_1$ and $\alpha \in \bar{B}$ and $\beta \in \bar{C}$. Thus, $(l(\alpha), l(\beta)) \in ser$ and $\neg(\alpha \diamond_s \beta)$. Hence, it follows from (11.3) that $\exists \delta, \gamma \in \Sigma_s$.

$\left(\begin{array}{l} pos_s(\delta) < pos_s(\gamma) \wedge (l(\delta), l(\gamma)) \notin ser \\ \alpha \sqsubset_s^* \delta \sqsubset_s^* \beta \wedge \alpha \sqsubset_s^* \gamma \sqsubset_s^* \beta \end{array} \right)$. By (2) and transitivity of \leq , we

have $\left(\begin{array}{l} \gamma \neq \delta \wedge (l(\delta), l(\gamma)) \notin ser \\ \wedge (\forall u \in [s]. pos_u(\alpha) \leq pos_u(\delta) \leq pos_u(\beta)) \\ \wedge (\forall u \in [s]. pos_u(\alpha) \leq pos_u(\gamma) \leq pos_u(\beta)) \end{array} \right)$. But since $\alpha, \beta \in \bar{B} \cup \bar{C} = \bar{A}$,

it follows that $\{\gamma, \delta\} \subseteq \bar{A}$, which implies $pos_{u_2}(\gamma) = pos_{u_2}(\delta)$. Since we also have $pos_s(\delta) < pos_s(\gamma)$, it follows that Proposition A.2(2) that there are z, w, D, E, F such that $\bar{z}\bar{D}\bar{w} \approx \bar{z}\bar{E}\bar{F}\bar{w}$ and $\delta \in \bar{E}$ and $\gamma \in \bar{F}$. Thus, $(l(\delta), l(\gamma)) \in ser$, a contradiction.

- If $pos_{u_1}(\alpha) < pos_{u_1}(\beta)$ and $pos_{u_2}(\alpha) > pos_{u_2}(\beta)$, then by Proposition A.2(1) there are x, y, A, B such that $\bar{u}_1 = \bar{x}\bar{A}\bar{B}\bar{y} (\approx \cup \approx^{-1}) \bar{x}\bar{B}\bar{A}\bar{y} = \bar{u}_2$ and $\alpha \in \bar{A}, \beta \in \bar{B}$. Thus, $(l(\alpha), l(\beta)) \in inl$. Since we assume $\alpha \prec_s \beta$, by (11.3), we have $(\alpha, \beta) \in \diamond_s \cap ((\sqsubset_s^*)^{\mathbb{M}} \circ \diamond_s^C \circ (\sqsubset_s^*)^{\mathbb{M}})$. So there are some γ, δ such that $\alpha (\sqsubset_s^*)^{\mathbb{M}} \gamma \diamond_s^C \delta (\sqsubset_s^*)^{\mathbb{M}} \beta$. Observe that

$$\begin{aligned}
& \alpha (\sqsubset_s^*)^{\mathbb{M}} \gamma \\
\implies & \alpha (\sqsubset_s^*)^{\mathbb{M}} \gamma \wedge \gamma (\sqsubset_s^*)^{\mathbb{M}} \alpha \\
\implies & \forall u \in [s]. pos_u(\alpha) \leq pos_u(\gamma) \wedge \forall u \in [s]. pos_u(\gamma) \leq pos_u(\alpha) \quad \langle \text{by (2)} \rangle \\
\implies & \forall u \in [s]. pos_u(\alpha) = pos_u(\gamma) \\
\implies & \{\alpha, \gamma\} \subseteq \bar{A} \quad \langle \text{since } \alpha \in \bar{A} \rangle
\end{aligned}$$

Similarly, since $\delta (\sqsubset_s^*)^{\mathbb{M}} \beta$, we can show that $\{\delta, \beta\} \subseteq \bar{B}$. Since $\bar{x}\bar{A}\bar{B}\bar{y} (\approx \cup \approx^{-1}) \bar{x}\bar{B}\bar{A}\bar{y}$, we get $A \times B \subseteq inl$. So $(l(\gamma), l(\delta)) \in inl$. But $\gamma \diamond_s^C \delta$ implies that $(l(\gamma), l(\delta)) \notin inl$, a contradiction. \square

Proposition A.4. Let s be a step sequence over a g -comtrace alphabet (E, sim, ser, inl) and $G^{\{s\}} = (\Sigma_s, \diamond, \sqsubset)$. If $\alpha, \beta \in \Sigma_s$, then

1. $\alpha \diamond \beta \implies \forall u \in [s]. pos_u(\alpha) \neq pos_u(\beta)$
2. $\alpha \sqsubset \beta \implies (\alpha \neq \beta \wedge \forall u \in [s]. pos_u(\alpha) \leq pos_u(\beta))$

PROOF. Follows from Definitions 11.3 and 11.1 and Proposition A.3. \square

Definition A.1 (serializable and non-serializable steps). Let A be a step over a g -comtrace alphabet (E, sim, ser, inl) and let $a \in A$ then:

1. Step A is called *serializable* iff

$$\exists B, C \in \widehat{\mathcal{P}}(A). B \cup C = A \wedge B \times C \subseteq ser.$$

Step A is called *non-serializable* iff A is not serializable. (Note that every non-serializable step is a synchronous step as defined in Definition 5.3.)

2. Step A is called *serializable to the left of a* iff

$$\exists B, C \in \widehat{\mathcal{P}}(A). B \cup C = A \wedge a \in B \wedge B \times C \subseteq ser.$$

Step A is called *non-serializable to the left of a* iff A is not serializable to the left of a , i.e., $\forall B, C \in \widehat{\mathcal{P}}(A). (B \cup C = A \wedge a \in B) \implies B \times C \not\subseteq ser$.

3. Step A is called *serializable to the right of a* iff

$$\exists B, C \in \widehat{\mathcal{P}}(A). B \cup C = A \wedge a \in C \wedge B \times C \subseteq ser.$$

Step A is called *non-serializable to the right of a* iff A is not serializable to the right of a , i.e., $\forall B, C \in \widehat{\mathcal{P}}(A). (B \cup C = A \wedge a \in C) \implies B \times C \not\subseteq ser$. \blacksquare

Proposition A.5. Let A be a step over a g -comtrace alphabet (E, sim, ser, inl) . Then

1. If A is non-serializable to the left of $l(\alpha)$ for some $\alpha \in \bar{A}$, then $\alpha \sqsubset_A^* \beta$ for all $\beta \in \bar{A}$.

2. If A is non-serializable to the right of $l(\beta)$ for some $\beta \in \bar{A}$, then $\alpha \sqsubset_A^* \beta$ for all $\alpha \in \bar{A}$.
3. If A is non-serializable, then $\forall \alpha, \beta \in \bar{A}. \alpha \sqsubset_A^* \beta$.

Before we proceed with the proof, observe that for all $\alpha, \beta \in \bar{A}$, $(l(\alpha), l(\beta)) \notin \text{inl}$. Hence, by Definition 11.3, we have

$$\alpha \sqsubset_A \beta \iff \text{pos}_A(\alpha) \leq \text{pos}_A(\beta) \wedge (l(\beta), l(\alpha)) \notin \text{ser}.$$

PROOF. 1. For any $\beta \in \bar{A}$, we have to show that $\alpha \sqsubset_A^* \beta$. We define the \sqsubset_A -right closure set of α inductively as follows:

$$RC^0(\alpha) \stackrel{\text{df}}{=} \{\alpha\} \quad RC^n(\alpha) \stackrel{\text{df}}{=} \{\delta \in \bar{A} \mid \exists \gamma \in RC^{n-1}(\alpha) \wedge \gamma \sqsubset_A \delta\}$$

We want to prove that if $\bar{A} \setminus RC^n(\alpha) \neq \emptyset$ then $|RC^{n+1}(\alpha)| > |RC^n(\alpha)|$. Assume that $\bar{A} \setminus RC^n(\alpha) \neq \emptyset$. Since $\alpha \in \bar{A}$ and A is non-serializable to the left of $l(\alpha)$, $l[\bar{A} \setminus RC^n(\alpha)] \times l[RC^n(\alpha)] \not\subseteq \text{ser}$. Thus there exists some $\gamma \in \bar{A} \setminus RC^n(\alpha)$ such that there is some $\delta \in RC^n(\alpha)$ satisfying $(l(\gamma), l(\delta)) \notin \text{ser}$. This implies $\delta \sqsubset_A \gamma$. Thus, $\gamma \in RC^{n+1}(\alpha)$ where $\gamma \notin RC^n(\alpha)$. So $|RC^{n+1}(\alpha)| > |RC^n(\alpha)|$ as desired.

Since A is finite and if $\bar{A} \setminus RC^n(\alpha) \neq \emptyset$ then $|RC^{n+1}(\alpha)| > |RC^n(\alpha)|$, for some $n < |A|$, we must have $RC^n(\alpha) = \bar{A}$. Thus, $\beta \in RC^n(\alpha)$. By the way the $RC^n(\alpha)$ is defined, we have $\alpha \sqsubset_A^* \beta$.

2. Dually to (1).

3. Since A is non-serializable, it follows that A is non-serializable to the left of $l(\alpha)$ for every $\alpha \in \bar{A}$. Hence, for every $\alpha \in \bar{A}$, we have $\forall \beta \in \bar{A}. \alpha \sqsubset_A^* \beta$. \square

The existence of a non-serializable sub-step of a step A to the left/right of an element $a \in A$ can be explained by the following proposition.

Proposition A.6. *Let A be a step over a g-comtrace alphabet $\Theta = (E, \text{sim}, \text{ser}, \text{inl})$ and $a \in A$. Then*

1. *There exists a unique $B \subseteq A$ such that $a \in B$, B is non-serializable to the left of a , and $A \neq B \implies A \equiv (A \setminus B)B$.*
2. *There exists a unique $C \subseteq A$ such that $a \in C$, C is non-serializable to the right of a , and $A \neq C \implies A \equiv C(A \setminus C)$.*
3. *There exists a unique $D \subseteq A$ such that $a \in D$, D is non-serializable, and $A \equiv xDy$, where x and y are step sequences over Θ .*

PROOF. 1. If A is non-serializable to the left of a , then $B = A$. If A is serializable to the left of a , then the following set is not empty:

$$\zeta \stackrel{\text{df}}{=} \{D \in \widehat{\mathcal{P}}(A) \mid \exists C \in \widehat{\mathcal{P}}(A). (C \cup D = A \wedge a \in D \wedge C \times D \subseteq \text{ser})\}$$

Let $B \in \zeta$ such that B is a minimal element of the poset (ζ, \subseteq) . We claim that B is non-serializable to the left of a . Suppose that B is serializable to the left of a , then there are some sets $E, F \in \widehat{\mathcal{P}}(B)$ such that $E \cup F = B \wedge a \in F \wedge E \times F \subseteq \text{ser}$. Since $B \in \zeta$, there is some set $H \in \widehat{\mathcal{P}}(A)$ such that $H \cup B = A \wedge a \in B \wedge H \times B \subseteq \text{ser}$. Because $H \times B \subseteq \text{ser}$ and $F \subseteq$

B , it follows that $H \times F \subseteq \text{ser}$. But since $E \times F \subseteq \text{ser}$, we have $(H \cup E) \times F \subseteq \text{ser}$. Hence, $(H \cup E) \cup F = A \wedge a \in F \wedge (H \cup E) \times F \subseteq \text{ser}$. So $E \in \zeta$ and $E \subset B$. This contradicts that B is minimal. Hence, B is non-serializable to the left of a .

By the way the set ζ is defined, $A \equiv (A \setminus B)B$. It remains to prove the uniqueness of B . Let $B' \in \zeta$ such that B' is a minimal element of the poset (ζ, \subset) . We want to show that $B = B'$.

We first show that $B \subseteq B'$. Suppose that there is some $b \in B$ such that $b \neq a$ and $b \notin B'$. Let α and β denote the event occurrences $a^{(1)}$ and $b^{(1)}$ in Σ_A respectively. Since $a \in B$ and B is non-serializable to the left of a , it follows from Proposition A.5(1) that $\alpha \sqsubset_A^* \beta$. But since $a \neq b$, $\alpha \sqsubset_A^* \setminus id_{\Sigma_A} \beta$. From the definition of \diamond -closure, it follows that $\alpha \sqsubset_{[A]} \beta$. Hence, by Proposition A.3(2), we have

$$\forall u \in [A]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta) \quad (\text{A.1})$$

By the way B' is chosen, we know $A \equiv (A \setminus B')B'$ and $b \notin B'$. So it follows that $b \in (A \setminus B')$. Hence, we have $(A \setminus B')B' \in [A]$ and $\text{pos}_{(A \setminus B')B'}(\beta) < \text{pos}_{(A \setminus B')B'}(\alpha)$, which contradicts (A.1). Thus, $B \subseteq B'$.

By reversing the role of B and B' , we can prove that $B \supseteq B'$. Hence, $B = B'$.

2. Dually to (1).

3. By (1) and (2), we only need to choose D such that D is non-serializable to the left and to the right of a . \square

Proposition A.7. *Let s be a step sequence over a g -comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$ and $G^{\{s\}} = (\Sigma_s, \diamond, \sqsubset)$. Let $\prec = \sqsubset \cup \diamond$. If $\alpha, \beta \in \Sigma_s$, then*

1. $\left(\begin{array}{l} (\forall u \in [s]. \text{pos}_u(\alpha) \neq \text{pos}_u(\beta)) \\ \wedge (\exists u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)) \\ \wedge (\exists u \in [s]. \text{pos}_u(\alpha) > \text{pos}_u(\beta)) \end{array} \right) \implies \alpha \diamond \beta$
2. $(\forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)) \implies \alpha \prec \beta$
3. $(\alpha \neq \beta \wedge \forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta)) \implies \alpha \sqsubset \beta$

PROOF. 1. If $\left(\begin{array}{l} (\forall u \in [s]. \text{pos}_u(\alpha) \neq \text{pos}_u(\beta)) \\ \wedge (\exists u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)) \\ \wedge (\exists u \in [s]. \text{pos}_u(\alpha) > \text{pos}_u(\beta)) \end{array} \right)$, then by Proposition A.2(1), there are $u_1, u_2 \in [s]$ and x, y, A, B such that $\overline{u_1} = \overline{x} \overline{A} \overline{B} \overline{y} (\approx \cup \approx^{-1}) \overline{x} \overline{B} \overline{A} \overline{y} = \overline{u_2}$ and $\alpha \in \overline{A}, \beta \in \overline{B}$. Hence, $(l(\alpha), l(\beta)) \in \text{inl}$, which by (11.1) implies that $\alpha \diamond_s \beta$. It then follows from Definitions 11.1 and 11.3 that $\alpha \diamond \beta$.

2, 3. Assume $\forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta)$ and $\alpha \neq \beta$. Hence, we can choose $u_0 \in [s]$ where $\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0}$ ($k \geq 1$), E_1, E_k are non-serializable, $\alpha \in \overline{E_1}, \beta \in \overline{E_k}$, and

$$\forall u'_0 \in [s]. \left(\begin{array}{l} \left(\overline{u'_0} = \overline{x'_0} \overline{E'_1} \dots \overline{E'_k} \overline{y'_0} \wedge \alpha \in \overline{E'_1} \wedge \beta \in \overline{E'_k} \right) \\ \implies \text{weight}(\overline{E_1} \dots \overline{E_k}) \leq \text{weight}(\overline{E'_1} \dots \overline{E'_k}) \end{array} \right) \quad (\text{A.2})$$

We will prove by induction on $\text{weight}(\overline{E_1} \dots \overline{E_k})$ that

$$(\forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)) \implies \alpha \prec \beta \quad (\text{A.3})$$

$$(\alpha \neq \beta \wedge \forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta)) \implies \alpha \sqsubset \beta \quad (\text{A.4})$$

Base case. When $\text{weight}(\overline{E_1} \dots \overline{E_k}) = 2$, then we consider two cases:

- If $\alpha \neq \beta$, $\forall u \in [s]$. $\text{pos}_u(\alpha) \leq \text{pos}_u(\beta)$ and $\exists u \in [s]$. $\text{pos}_u(\alpha) = \text{pos}_u(\beta)$, then it follows that

- $\overline{u_0} = \overline{x_0}\{\alpha, \beta\}\overline{y_0}$, or
- $\overline{u_0} = \overline{x_0}\{\alpha\}\{\beta\}\overline{y_0} \equiv \overline{x_0}\{\alpha, \beta\}\overline{y_0}$

But since $\forall u \in [s]$. $\text{pos}_u(\alpha) \leq \text{pos}_u(\beta)$, in either case, we must have $\{l(\alpha), l(\beta)\}$ is not serializable to the right of $l(\beta)$. Hence, by Proposition A.5(2), $\alpha \sqsubset_s^* \beta$. This by Definitions 11.1 and 11.3 implies that $\alpha \sqsubset \beta$.

- If $\forall u \in [s]$. $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$, then it follows $\overline{u_0} = \overline{x_0}\{\alpha\}\{\beta\}\overline{y_0}$. Since we assume that $\forall u \in [s]$. $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$, we must have $(l(\alpha), l(\beta)) \notin \text{ser} \cup \text{inl}$. This, by (11.1), implies that $\alpha \prec_s \beta$. Hence, from Definitions 11.1 and 11.3, we get $\alpha \prec \beta$.

Since $\prec \subseteq \sqsubset$, it follows from these two cases that (A.3) and (A.4) hold.

Inductive step. When $\text{weight}(\overline{E_1} \dots \overline{E_k}) > 2$, then $\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0}$ where $k \geq 1$. We need to consider two cases:

Case (i): If $\alpha \neq \beta$ and $\forall u \in [s]$. $\text{pos}_u(\alpha) \leq \text{pos}_u(\beta)$ and $\exists u \in [s]$. $\text{pos}_u(\alpha) = \text{pos}_u(\beta)$, then there is some v_0 $\overline{v_0} = \overline{w_0} \overline{E} \overline{z_0}$ and $\alpha, \beta \in \overline{E}$. Either E is non-serializable to the right of $l(\beta)$, or by Proposition A.6(2) $\overline{v_0} = \overline{w_0} \overline{E} \overline{z_0} \equiv \overline{w'_0} \overline{E'} \overline{z'_0}$ where E' is non-serializable to the right of $l(\beta)$. In either case, by Proposition A.5(2), we have $\alpha \sqsubset_s^* \beta$. So it follows from Definitions 11.1 and 11.3 that $\alpha \sqsubset \beta$.

Case (ii): If $\forall u \in [s]$. $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$, then it follows $\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0}$ where $k \geq 2$ and $\alpha \in \overline{E_1}, \beta \in \overline{E_k}$. If $(l(\alpha), l(\beta)) \notin \text{ser} \cup \text{inl}$, then by (11.1), $\alpha \prec_s \beta$. Hence, from Definitions 11.1 and 11.3, we get $\alpha \prec \beta$. So we need to consider only when $(l(\alpha), l(\beta)) \in \text{ser}$ or $(l(\alpha), l(\beta)) \in \text{inl}$. There are three cases to consider:

- If $\overline{u_0} = \overline{x_0} \overline{E_1} \overline{E_2} \overline{y_0}$ where E_1 and E_2 are non-serializable, then since we assume $\forall u \in [s]$. $\text{pos}_u(\alpha) < \text{pos}_u(\beta)$, it follows that $E_1 \times E_2 \not\subseteq \text{ser}$ and $E_1 \times E_2 \not\subseteq \text{inl}$. Hence, there are $\alpha_1, \alpha_2 \in \overline{E_1}$ and $\beta_1, \beta_2 \in \overline{E_2}$ such that $(l(\alpha_1), l(\beta_1)) \notin \text{inl}$ and $(l(\alpha_2), l(\beta_2)) \notin \text{ser}$. Since E_1 and E_2 are non-serializable, by Proposition A.5(3), $\alpha_1 \sqsubset_s^* \alpha_2$ and $\beta_2 \sqsubset_s^* \beta_1$. Also by 11.3, we know that $\alpha_1 \diamond_s \beta_2$ and $\alpha_2 \diamond_s^C \beta_1$. Thus, by 11.3, we have $\alpha_1 \prec_s \beta_2$. Since E_1 and E_2 are non-serializable, by Proposition A.5(3), $\alpha \sqsubset_s^* \alpha_1 \prec_s \beta_2 \sqsubset_s^* \beta$. Hence, by Definitions 11.1 and 11.3, $\alpha \prec \beta$.
- If $\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0}$ where $k \geq 3$ and $(l(\alpha), l(\beta)) \in \text{inl}$, then let $\gamma \in \overline{E_2}$. Observe that we must have

$$\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0} \equiv \overline{x_1} \overline{E_1} \overline{w_1} \overline{F} \overline{z_1} \overline{E_k} \overline{y_1} \equiv \overline{x_2} \overline{E_1} \overline{w_2} \overline{F} \overline{z_2} \overline{E_k} \overline{y_2}$$

such that $\gamma \in \overline{F}$, F is a non-serializable, and $\text{weight}(\overline{E_1} \overline{w_1} \overline{F}), \text{weight}(\overline{F} \overline{z_2} \overline{E_k})$ satisfy the minimal condition similarly to (A.2). Since from the way u_0 is chosen, we know that $\forall u \in [s]$. $\text{pos}_u(\alpha) \leq \text{pos}_u(\gamma)$ and $\forall u \in [s]$. $\text{pos}_u(\gamma) \leq \text{pos}_u(\beta)$, by applying the induction hypothesis, we get

$$\alpha \sqsubset \gamma \sqsubset \beta \tag{A.5}$$

So by transitivity of \sqsubset , we get $\alpha \sqsubset \beta$. But since we assume $(l(\alpha), l(\beta)) \in \text{inl}$, it follows that $\alpha \diamond \beta$. Hence, $(\alpha, \beta) \in \sqsubset \cap \diamond = \prec$.

(c) If $\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0}$ where $k \geq 3$ and $(l(\alpha), l(\beta)) \in \text{ser}$, then we observe from how u_0 is chosen that

$$\forall \gamma \in \biguplus(\overline{E_1} \dots \overline{E_k}). (\forall u \in [s]. \text{pos}_{u_0}(\alpha) \leq \text{pos}_{u_0}(\gamma) \leq \text{pos}_{u_0}(\beta))$$

Similarly to how we show (A.5), we can prove that

$$\forall \gamma \in \biguplus(\overline{E_1} \dots \overline{E_k}) \setminus \{\alpha, \beta\}. \alpha \sqsubset \gamma \sqsubset \beta \quad (\text{A.6})$$

We next want to show that

$$\exists \delta, \gamma \in \biguplus(\overline{E_1} \dots \overline{E_k}). (\text{pos}_{u_0}(\delta) < \text{pos}_{u_0}(\gamma) \wedge (l(\delta), l(\gamma)) \notin \text{ser}) \quad (\text{A.7})$$

Suppose that (A.7) does not hold, then

$$\forall \delta, \gamma \in \biguplus(\overline{E_1} \dots \overline{E_k}). (\text{pos}_{u_0}(\delta) < \text{pos}_{u_0}(\gamma) \implies (l(\delta), l(\gamma)) \in \text{ser})$$

It follows that $\overline{u_0} = \overline{x_0} \overline{E_1} \dots \overline{E_k} \overline{y_0} \equiv \overline{x_0} \overline{E} \overline{y_0}$, which contradicts that $\forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)$. Hence, we have shown (A.7).

Let $\delta, \gamma \in \biguplus(\overline{E_1} \dots \overline{E_k})$ be event occurrences such that $\text{pos}_{u_0}(\delta) < \text{pos}_{u_0}(\gamma)$ and $(l(\delta), l(\gamma)) \notin \text{ser}$. By (A.6), $\alpha(\sqsubset \cup id_{\Sigma_s})\delta(\sqsubset \cup id_{\Sigma_s})\beta$ and $\alpha(\sqsubset \cup id_{\Sigma_s})\gamma(\sqsubset \cup id_{\Sigma_s})\beta$. If $\alpha \prec \delta$ or $\delta \prec \beta$ or $\alpha \prec \gamma$ or $\gamma \prec \beta$, then by (C4) of Definition 4.1, $\alpha \prec \beta$. Otherwise, by Definitions 11.1 and 11.3, we have $\alpha \sqsubset_s^* \delta \sqsubset_s^* \beta$ and $\alpha \sqsubset_s^* \gamma \sqsubset_s^* \beta$. But since $\text{pos}_{u_0}(\delta) < \text{pos}_{u_0}(\gamma)$ and $(l(\delta), l(\gamma)) \notin \text{ser}$, by Definition 11.3, $\alpha \prec_s \beta$. So by Definitions 11.1 and 11.3, $\alpha \prec \beta$.

Thus, we have shown (A.3) and (A.4) as desired. \square

Proposition 11.4. *Let s be a step sequence over a g-comtrace alphabet $(E, \text{sim}, \text{ser}, \text{inl})$. Let $G^{\{s\}} = (\Sigma_s, \diamond, \sqsubset)$, and let $\prec = \diamond \cap \sqsubset$. Then for every $\alpha, \beta \in \Sigma_s$, we have*

1. $\alpha \diamond \beta \iff \forall u \in [s]. \text{pos}_u(\alpha) \neq \text{pos}_u(\beta)$
2. $\alpha \sqsubset \beta \iff (\forall u \in [s]. \text{pos}_u(\alpha) \leq \text{pos}_u(\beta) \wedge \alpha \neq \beta)$
3. $\alpha \prec \beta \iff \forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)$
4. If $l(\alpha) = l(\beta)$ and $\text{pos}_s(\alpha) < \text{pos}_s(\beta)$, then $\alpha \prec \beta$.

PROOF. 1. Follows from Proposition A.4(1) and Proposition A.7(1, 2).

2. Follows from Proposition A.4(2) and Proposition A.7(3).

3. Follows from (1) and (2).

4. Assume that $l(\alpha) = l(\beta)$ and $\text{pos}_s(\alpha) < \text{pos}_s(\beta)$. Then by Proposition A.1(2), we know $\forall u \in [s]. \text{pos}_u(\alpha) < \text{pos}_u(\beta)$. Hence, by (3), $\alpha \prec \beta$. \square

Appendix B: Proof of Lemma 11.2.

Lemma 11.2. *Let $s, t \in \mathbb{S}^*$ and $\triangleleft_s \in \text{ext}(G^{\{t\}})$. Then $G^{\{s\}} = G^{\{t\}}$.*

PROOF. ($\diamond_t = \diamond_s$) We have $\alpha \diamond_t \beta$ iff by Definition 11.3 $(l(\alpha), l(\beta)) \in \text{inl}$, which by Definition 11.3 means $\alpha \diamond_s \beta$. Hence,

$$\diamond_t = \diamond_s \quad (\text{B.8})$$

($\sqsubset_t = \sqsubset_s$) If $\alpha \sqsubset_t \beta$, then by Definitions 11.1 and 11.3, $\alpha \sqsubset_s \beta$. But since $\triangleleft_s \in \text{ext}(G^{\{t\}})$, we have $\alpha \triangleleft_s \beta$, which implies $\text{pos}_s(\alpha) \leq \text{pos}_s(\beta)$. But since $\alpha \sqsubset_t \beta$, by Definition 11.3, $(l(\beta), l(\alpha)) \notin \text{ser} \cup \text{inl}$. Hence, by Definition 11.3, $\alpha \sqsubset_s \beta$. Thus,

$$\sqsubset_t \subseteq \sqsubset_s \quad (\text{B.9})$$

It remains to show that $\sqsubset_s \subseteq \sqsubset_t$. Let $\alpha \sqsubset_s \beta$, and we suppose that $\neg(\alpha \sqsubset_t \beta)$. Since $\alpha \sqsubset_s \beta$, by Definition 11.3, $\text{pos}_s(\alpha) \leq \text{pos}_s(\beta)$ and $(l(\beta), l(\alpha)) \notin \text{ser} \cup \text{inl}$. Since we assume $\neg(\alpha \sqsubset_t \beta)$, by Definition 11.3, we must have $\text{pos}_t(\beta) < \text{pos}_t(\alpha)$. Hence, by Definitions 11.1 and 11.3, $\beta \prec_t \alpha$ and $\beta \prec \alpha$. But since $\triangleleft_s \in \text{ext}(G^{\{t\}})$, we have $\beta \triangleleft_s \alpha$. So $\text{pos}_s(\beta) < \text{pos}_s(\alpha)$, a contradiction. Thus, $\sqsubset_s \subseteq \sqsubset_t$. Together with (B.9), we get

$$\sqsubset_t = \sqsubset_s \quad (\text{B.10})$$

($\prec_t = \prec_s$) If $\alpha \prec_t \beta$, then by Definitions 11.1 and 11.3, $\alpha \prec \beta$. But since $\triangleleft_s \in \text{ext}(G^{\{t\}})$, we have $\alpha \triangleleft_s \beta$, which implies

$$\text{pos}_s(\alpha) < \text{pos}_s(\beta) \quad (\text{B.11})$$

Since $\alpha \prec_t \beta$, by Definition 11.3, we have

$$\begin{aligned} & (l(\alpha), l(\beta)) \notin \text{ser} \cup \text{inl} \\ \vee & (\alpha, \beta) \in \diamond_t \cap ((\sqsubset_t^*)^{\text{m}} \circ \diamond_t^{\text{C}} \circ (\sqsubset_t^*)^{\text{m}}) \\ \vee & \left(\begin{array}{l} (l(\alpha), l(\beta)) \in \text{ser} \\ \wedge \exists \delta, \gamma \in \Sigma_t. \left(\begin{array}{l} \text{pos}_t(\delta) < \text{pos}_t(\gamma) \wedge (l(\delta), l(\gamma)) \notin \text{ser} \\ \alpha \sqsubset_t^* \delta \sqsubset_t^* \beta \wedge \alpha \sqsubset_t^* \gamma \sqsubset_t^* \beta \end{array} \right) \end{array} \right) \end{aligned}$$

We want to show that $\alpha \prec_s \beta$. There are three cases to consider:

- (a) When $(l(\alpha), l(\beta)) \notin \text{ser} \cup \text{inl}$, it follows from (B.11) and Definition 11.3 that $\alpha \prec_s \beta$.
- (b) When $(\alpha, \beta) \in \diamond_t \cap ((\sqsubset_t^*)^{\text{m}} \circ \diamond_t^{\text{C}} \circ (\sqsubset_t^*)^{\text{m}})$, then $\alpha \diamond_t \beta$ and there are $\delta, \gamma \in \Sigma$ such that $\alpha \sqsubset_t^* \delta \diamond_t^{\text{C}} \gamma \sqsubset_t^* \beta$. Since $\sqsubset_t = \sqsubset_s$ and $\diamond_t = \diamond_s$, we have $\alpha \diamond_s \beta$ and $\alpha \sqsubset_s^* \delta \diamond_s^{\text{C}} \gamma \sqsubset_s^* \beta$. Thus, it follows from (B.11) and Definition 11.3 that $\alpha \prec_s \beta$.
- (c) There remains only the case when $(l(\alpha), l(\beta)) \in \text{ser}$ and there are $\delta, \gamma \in \Sigma_t$ such that $\left(\begin{array}{l} \text{pos}_t(\delta) < \text{pos}_t(\gamma) \wedge (l(\delta), l(\gamma)) \notin \text{ser} \\ \wedge \alpha \sqsubset_t^* \delta \sqsubset_t^* \beta \wedge \alpha \sqsubset_t^* \gamma \sqsubset_t^* \beta \end{array} \right)$. Since $\sqsubset_t = \sqsubset_s$, we also have $\alpha \sqsubset_s^* \delta \sqsubset_s^* \beta \wedge \alpha \sqsubset_s^* \gamma \sqsubset_s^* \beta$. Since $(l(\delta), l(\gamma)) \notin \text{ser}$, we either have $(l(\delta), l(\gamma)) \in \text{inl}$ or $(l(\delta), l(\gamma)) \notin \text{ser} \cup \text{inl}$.
 - If $(l(\delta), l(\gamma)) \in \text{inl}$, then $\text{pos}_s(\delta) \neq \text{pos}_s(\gamma)$. This implies $(\text{pos}_s(\delta) < \text{pos}_s(\gamma) \wedge (l(\delta), l(\gamma)) \notin \text{ser})$ or $(\text{pos}_s(\gamma) < \text{pos}_s(\delta) \wedge (l(\gamma), l(\delta)) \notin \text{ser})$. So it follows from (B.11) and Definition 11.3 that $\alpha \prec_s \beta$.
 - If $(l(\delta), l(\gamma)) \notin \text{inl}$, then $(l(\delta), l(\gamma)) \notin \text{ser} \cup \text{inl}$. Hence, by Definition 11.3, $\delta \prec_t \gamma$, which by Definitions 11.1 and 11.3, $\delta \prec \gamma$. But since $\triangleleft_s \in \text{ext}(G^{\{t\}})$, we have $\delta \triangleleft_s \gamma$, which implies $\text{pos}_s(\delta) < \text{pos}_s(\gamma)$. Since $\text{pos}_s(\delta) < \text{pos}_s(\gamma)$ and $(l(\delta), l(\gamma)) \notin \text{ser}$, it follows from (B.11) and Definition 11.3 that $\alpha \prec_s \beta$.

Thus, we have shown that $\alpha \prec_s \beta$. Hence,

$$\prec_t \subseteq \prec_s \quad (\text{B.12})$$

It remains to show that $\prec_s \subseteq \prec_t$. Let $\alpha \prec_s \beta$. Suppose that $\neg(\alpha \prec_t \beta)$. Since $\alpha \prec_s \beta$, by Definition 11.3, $pos_s(\alpha) < pos_s(\beta)$ and

$$\left(\begin{array}{l} (l(\alpha), l(\beta)) \notin ser \cup inl \\ \vee (\alpha, \beta) \in \diamond_s \cap ((\sqsubset_s^*)^{\mathbb{M}} \circ \diamond_s^C \circ (\sqsubset_s^*)^{\mathbb{M}}) \\ \vee \left(\begin{array}{l} (l(\alpha), l(\beta)) \in ser \\ \wedge \exists \delta, \gamma \in \Sigma_s. \left(\begin{array}{l} pos_s(\delta) < pos_s(\gamma) \wedge (l(\delta), l(\gamma)) \notin ser \\ \alpha \sqsubset_s^* \delta \sqsubset_s^* \beta \wedge \alpha \sqsubset_s^* \gamma \sqsubset_s^* \beta \end{array} \right) \end{array} \right) \end{array} \right).$$

We want to show that $\alpha \prec_t \beta$. We consider three cases:

- (a) When $(l(\alpha), l(\beta)) \notin ser \cup inl$, we suppose that $\neg(\alpha \prec_t \beta)$. This by Definition 11.3 implies that $pos_t(\beta) \leq pos_t(\alpha)$. By Definitions 11.1 and 11.3, it follows that $\beta \sqsubset_t \alpha$ and $\beta \sqsubset \alpha$. But since $\triangleleft_s \in ext(G^{\{t\}})$, we have $\beta \triangleleft_s \alpha$, which implies $pos_s(\beta) \leq pos_s(\alpha)$, a contradiction.
- (b) If $(\alpha, \beta) \in \diamond_s \cap ((\sqsubset_s^*)^{\mathbb{M}} \circ \diamond_s^C \circ (\sqsubset_s^*)^{\mathbb{M}})$, then since $\diamond_s = \diamond_t$ and $\sqsubset_s = \sqsubset_t$, we have $(\alpha, \beta) \in \diamond_t \cap ((\sqsubset_t^*)^{\mathbb{M}} \circ \diamond_t^C \circ (\sqsubset_t^*)^{\mathbb{M}})$. Since $\alpha \prec_t \beta$, we have $pos_t(\alpha) < pos_t(\beta)$ or $pos_t(\beta) < pos_t(\alpha)$. We claim that $pos_t(\alpha) < pos_t(\beta)$. Suppose for a contradict that $pos_t(\beta) < pos_t(\alpha)$. Since $(\alpha, \beta) \in \diamond_t \cap ((\sqsubset_t^*)^{\mathbb{M}} \circ \diamond_t^C \circ (\sqsubset_t^*)^{\mathbb{M}})$ and \diamond_t is symmetric, we have $(\beta, \alpha) \in \diamond_t \cap ((\sqsubset_t^*)^{\mathbb{M}} \circ \diamond_t^C \circ (\sqsubset_t^*)^{\mathbb{M}})$. Hence, it follows from Definitions 11.1 and 11.3 that $\beta \prec_t \alpha$ and $\beta \prec \alpha$. But since $\triangleleft_s \in ext(G^{\{t\}})$, we have $\beta \triangleleft_s \alpha$, which implies $pos_s(\beta) < pos_s(\alpha)$, a contradiction. We have just shown that $pos_t(\alpha) < pos_t(\beta)$. Since $(\alpha, \beta) \in \diamond_t \cap ((\sqsubset_t^*)^{\mathbb{M}} \circ \diamond_t^C \circ (\sqsubset_t^*)^{\mathbb{M}})$, we get $\alpha \prec_t \beta$.
- (c) There remains only the case when $(l(\alpha), l(\beta)) \in ser$ and there are $\delta, \gamma \in \Sigma_s$ such that $\left(\begin{array}{l} pos_s(\delta) < pos_s(\gamma) \wedge (l(\delta), l(\gamma)) \notin ser \\ \wedge \alpha \sqsubset_s^* \delta \sqsubset_s^* \beta \wedge \alpha \sqsubset_s^* \gamma \sqsubset_s^* \beta \end{array} \right)$. Since $\sqsubset_s = \sqsubset_t$, we have $\alpha \sqsubset_t^* \delta \sqsubset_t^* \beta$ and $\alpha \sqsubset_t^* \gamma \sqsubset_t^* \beta$, which by Definition 11.3 and transitivity of \leq implies that $pos_t(\alpha) \leq pos_t(\delta) \leq pos_t(\beta)$ and $pos_t(\alpha) \leq pos_t(\gamma) \leq pos_t(\beta)$. Since $(l(\delta), l(\gamma)) \notin ser$, we either have $(l(\delta), l(\gamma)) \in inl$ or $(l(\delta), l(\gamma)) \notin ser \cup inl$.
 - (i) If $(l(\delta), l(\gamma)) \in inl$, then $pos_t(\delta) \neq pos_t(\gamma)$. This implies that $(pos_t(\delta) < pos_t(\gamma) \wedge (l(\delta), l(\gamma)) \notin ser)$ or $(pos_t(\gamma) < pos_t(\delta) \wedge (l(\gamma), l(\delta)) \notin ser)$. Since $pos_t(\delta) \neq pos_t(\gamma)$ and $pos_t(\alpha) \leq pos_t(\delta) \leq pos_t(\beta)$ and $pos_t(\alpha) \leq pos_t(\gamma) \leq pos_t(\beta)$, we also have $pos_t(\alpha) < pos_t(\beta)$. So it follows from Definition 11.3 that $\alpha \prec_t \beta$.
 - (ii) If $(l(\delta), l(\gamma)) \notin inl$, then $(l(\delta), l(\gamma)) \notin ser \cup inl$. We want to show that $pos_t(\delta) < pos_t(\gamma)$. Suppose that $pos_s(\delta) \geq pos_s(\gamma)$. Since $(l(\delta), l(\gamma)) \notin ser \cup inl$, by Definitions 11.1 and 11.3, we have $\gamma \sqsubset_t \delta$ and $\gamma \sqsubset \delta$. But since $\triangleleft_s \in ext(G^{\{t\}})$, we have $\gamma \triangleleft_s \delta$, which implies $pos_s(\gamma) \leq pos_s(\delta)$, a contradiction. Since $pos_t(\delta) < pos_t(\gamma)$ and $pos_t(\alpha) \leq pos_t(\delta) \leq pos_t(\beta)$ and $pos_t(\alpha) \leq pos_t(\gamma) \leq pos_t(\beta)$, we have $pos_t(\alpha) < pos_t(\beta)$. Hence, we have $pos_t(\alpha) < pos_t(\beta)$ and $\left(\begin{array}{l} pos_t(\delta) < pos_t(\gamma) \wedge (l(\delta), l(\gamma)) \notin ser \cup inl \\ \wedge \alpha \sqsubset_t^* \delta \sqsubset_t^* \beta \wedge \alpha \sqsubset_t^* \gamma \sqsubset_t^* \beta \end{array} \right)$. So it follows that $\alpha \prec_t \beta$ by Definition 11.3.

Thus, we have shown $\prec_s \subseteq \prec_t$. This and (B.12) imply

$$\prec_t = \prec_s \quad (\text{B.13})$$

By (B.8), (B.10) and (B.13), we have $(\Sigma, \diamond_t \cup \prec_t, \sqsubset_t \cup \prec_t) = (\Sigma, \diamond_s \cup \prec_s, \sqsubset_s \cup \prec_s)$. Thus, it follows that $G^{\{t\}} = (\Sigma, \diamond_t \cup \prec_t, \sqsubset_t \cup \prec_t)^{\boxtimes} = (\Sigma, \diamond_s \cup \prec_s, \sqsubset_s \cup \prec_s)^{\boxtimes} = G^{\{s\}}$. \square